Høyskolen Kristiania

Bachelorprosjekt vår 2021

Emnekode og emnenavn:	BAO301 Bachelorprosjekt
Tittel norsk:	Automatisering av møtenotater: Hvordan øke verdien av møter gjennom automatisk transkribering.
Tittel engelsk:	Automation of meeting notes: How to increase the value of meetings through automatic transcription.
Oppdragsgiver	Kodebyraaet AS
Innleveringsdato:	03.06.21
Antall sider:	89
Antall ord:	9543
Tilgjengelighet:	Fri

Sammendrag:

Denne rapporten handler om utviklingen av en løsning som automatiserer transkribering, med fokus på opptak fra digitale møter. Verktøyet gir også brukeren mulighet til å redigere teksten før de kan velge å dele teksten via e-post. Målet er å gjøre hverdagen enklere for de som bruker mye av tiden sin i møter ved å lage et verktøy som er raskt og enkelt å bruke. Ved å delta i møter uten å måtte notere er målet at deltakere bedre kan holde fokus på møtet. Utviklingen har bestått av en innsiktsfase, designsprinter, brukertesting, prototyping og koding av en minste fungerende løsning (MVP).

Gruppenummer:	36	
Studentnavn:	**Studentnummer:**	**Signatur:**
Bernt Johan Aspehaug	705372	
Martin Gundersby Molvær	705800	
Per-Kristian Vinje	705370	
Denne bacheloroppgaven er gjennomført som en del av utdannelsen ved Høyskolen Kristiania. Høyskolen er ikke ansvarlig for oppgavens metoder, resultater, konklusjoner eller anbefalinger.		

Innholdsfortegnelse

1. Innledning

For å kartlegge ambisjoner, og hvilke forventninger vi i gruppen hadde til hverandre skrev vi alle under på en gruppekontrakt (Se vedlegg A: Arbeidskontrakt). Vi går alle tre Frontend- og mobilutvikling, som gir oss et godt utgangspunkt for både teknisk løsning og design.

1.1. Presentasjon av oppdragsgiver

Kodebyraaet AS er er en bedrift som koder og utvikler nettsider, tjenester og produkter. De ble etablert i 2011. Bedriften har 18 ansatte og holder til i Ålesund, Oslo og Polen. De samarbeider med "noen av landets mest spennende selskaper og kreative byråer" som blant annet Snøhetta og Anti. I tillegg utvikler de egne prosjekter, deriblant billettsystemet Tikkio. De bruker designmetodikk og teknologi for å skape helhetlige produkter for sine kunder. I 2019 hadde de en omsetning på nesten 20 millioner kroner. Våre kontaktpersoner i Kodebyraaet var daglig leder Emil Bonsaksen og Johan Jøsok, COO (chief operating officer) (Kodebyraaet, 2021).

1.2. Beskrivelse av prosjektet

Vi startet med en plan om å lage en mobilapplikasjon som ga mulighet for å sette inn markører/kapitler i viktige deler av møteopptak hvor man er fysisk tilstede i samme rom. Etter litt over en uke med designsprinting, med mobilapplikasjonen som fokus, fant vi i møte med Kodebyraaet (heretter KB) ut at dette ikke lenger var like relevant. Møter foregår i dag i stor grad over nett. For å tilpasse oss denne endringen ble det bestemt at vi skulle dreie fokuset fra en mobilapplikasjon over til et nettleser-basert verktøy. For å dekke det samme behovet så vi på løsninger for å gjøre etterarbeid av møter enklere.

Etter å ha kommet frem til et design vi likte med mobilapplikasjonen som mål, var det først uvant å skulle legge fra seg dette og starte på nytt. Denne opplevelsen ser vi også på som viktig da det er lett å låse seg til noe og dermed gå glipp av en bedre løsning. Slik man arbeider i bransjen er det ikke uvanlig å måtte endre på planene underveis. Skal man lykkes må man være "agile", klare å gjøre endringer basert på kvalitative undersøkelser og tester, og stole på tilbakemeldingene man får.

1.3. Bedriftens ønsker

Ved oppstart av prosjektet hadde vi et møte med KB der vi fikk presentert oppgaven (se vedlegg B: Møtereferater). De ønsket opprinnelig en applikasjon som skulle bli et verktøy for å unngå notering underveis i møter. Hensikten skulle være at den kunne brukes uten å være forstyrrende og det skulle kunne settes markør/"kapittel" ved å tappe på telefonen når noe viktig i møtet ble tatt opp. På grunn av Covid-19-pandemien var ikke lenger fysiske møter og behovet for en slik mobilapplikasjon like relevant. Dermed endret KBs ønske seg til et verktøy med fokus på digitale møter. I videre samtaler kom vi sammen med KB fram til ideen om et verktøy for transkribering, redigering og deling av møteopptak.

Vi skulle utvikle et Minimum Viable Product (MVP), eller minste fungerende løsning, som de eventuelt kunne videreutvikle til et reelt produkt. En MVP er en versjon av et produkt med akkurat nok funksjonalitet til å kunne testes av potensielle kunder som kan bidra med tilbakemeldinger for videre utvikling (Wikipedia, 2021). Løsningen skal brukes for å transkribere møter og ha som mål å eliminere behovet for å notere underveis. På den måten kan man gi brukeren mulighet til å delta i møter uten distraksjon og pauser for å ta notater. Nøkkelfunksjonalitet skal inkludere transkribering av lyd- og videofiler, behandling av tekst for oppsummering og deling.

KB ønsket også at koden skulle være godt dokumentert og kommentert slik det ville være lett å ta over koden for eventuell videreutvikling. De ønsket også gode tester for å sikre at funksjonalitet fungerte som forventet.

1.4 Gruppens mål

Vi satte oss som mål å levere en løsning som ville tilfredsstille KBs visjon, samtidig som alle i gruppen fikk brukt egen kreativitet og kompetanse. Vi var opptatt av godt samarbeid hvor vi la vekt på prosess, utnyttelse av verktøy og metodikk for å sikre en relevant og realistisk arbeidshverdag under prosjektet. Ved å møte målene til KB kunne vi utvikle en løsning som bidrar til forenklet etterarbeid av opptak fra møter. Det var også en mulighet til å få et inntrykk av hverdagen i bransjen, og å kunne tilføre noe relevant på CV-en.

1.5. Problemstilling

Uten å bli for tilspisset kom vi frem til en problemstilling som samsvarte godt med oppdragsgivers ønsker:

Hvordan kan man gjøre opptak av digitale møter mer verdifulle?

Som nevnt under bedriftens ønsker er det ofte vendinger i slike prosjekter. Man finner nye behov og utfordringer underveis. Selv om fokuset skiftet fra mobilapplikasjon til en nettbasert løsning, var problemstillingen like relevant.

1.6. Teoretisk grunnlag til forskningsområde

Basert på bedriftens ønsker og egne diskusjoner så vi etter relevant forskning rundt temaer som transkribering, oppmerksomhet i møter, bruk av notater i etterkant av møter og bruk av verktøy for å ta og redigere notater. Det meste av litteraturen har vi funnet gjennom søk i Google Scholar, IEEE og ResearchGate.

Når et menneske transkriberer er det ikke bare en teknisk prosedyre. Det er en fortolkende handling. Siden det er vanskelig å tolke kompleksiteten i menneskelig kommunikasjon kan det utelates informasjon når man tolker en transkribert tekst. Ved å høre og/eller se på de originale opptakene, kan man bedre forstå hvordan noe har blitt sagt i tillegg til hva som har blitt sagt (Bailey, 2008). I motsetning til menneskelig transkribering tar ikke programvare høyde for mellommenneskelig kommunikasjon.

Programvare for stemmegjenkjenning er programvare som automatisk transkriberer digitale opptak uten behovet for å skrive. Det har vært tilgjengelig for allmennheten siden tidlig på 80-tallet. De nyeste versjonene kan ha opp til 98 % nøyaktighetsgrad – høyere enn mange menneskelige transkribenter kan skryte av. Videre har programvare utviklet seg de siste 20 årene fra å bare kunne forstå ett ord av gangen med pauser imellom, til å forstå kontinuerlig tale. Nyere versjoner kan ha vokabular på forskjellige språk og dialekter som kan endres etter behov (Matheson, 2015).

Motivasjonen for å se over møtenotater er variert og ikke helt klar. Richter et al. (2001, s. 123-128) viser til en studie fra 1994 der det ble utført intervjuer angående problemer med å ta notater underveis i møter. De fant flere utfordringer som opptak av møter kunne løse. Som å ikke få med seg viktige fakta, mangel på tid for å skrive ned alt, redusert deltagelse og mangel på tilstrekkelige notater for å senere forstå hva som hadde blitt skrevet. Opptak av møter kunne også bidra med å bevare informasjon som kunne gå tapt over tid. Selv om de fleste av dagens møteplattformer kan gjøre opptak, må noen fortsatt ta notater. Whittaker, Laban og Tucker (2006) fant ut at 63 % av deres deltakere i studien "Analysing Meeting Records: An Ethnographic Study and Technological Implications" alltid tok personlige notater i møter. De resterende 37 % gjorde det av og til og pekte på faktorer som for eksempel å være leder av møtet som forhindret dem i å ta egne notater. Alle deltakerne refererte tilbake til sine notater minst én gang, hvorav 75 % gjorde det ofte. Et annet nøkkelpunkt var at de passet godt på at notatene var nøyaktige. Halvparten skrev av og til om på notatene, og de så til å lagre notatene så lenge som et år i gjennomsnitt.

2. Bakgrunn

I denne delen tar vi for oss forventninger til løsningen, samt utformingen av den. På bakgrunn av brukertester, brukeropplevelser og etablert forskning, vil vi også legge frem hvordan vi har kommet frem til design og funksjonalitet.

2.1. Oppdragsgivers forventninger

På spørsmål om forventninger til oppgaven, svarte KB at de ville se en MVP. De ville ikke stille strenge krav til spesifikasjon, men heller la oss ta avgjørelsene om hva vi følte var relevant for en MVP. Det ble også opp til oss selv å finne ut hva slags teknologi som gjorde løsningen mulig. Vi fikk frihet til å være agile og det ble forventet at vi var fleksible og åpne for innspill underveis i prosjektet.

2.2. Innsikt

For å skape en god brukeropplevelse har vi benyttet forskjellige metoder for å forstå brukerne og deres behov. Forskjellige brukere har forskjellige behov. Det er derfor trolig en

god metode å lage personas som skal kunne formidle disse forskjellige behovene på en kort og enkel måte. Vi har utviklet tre personas, som skal representere våre brukeres behov (se vedlegg C: Personas). Videre har vi gjort litteratursøk med fokus på forskning om brukeratferd ved lasting i programvare.

2.2.1. Eksisterende løsninger

I oppstarten av prosjektet utformet vi litteratursøk for å skaffe innsikt i rammeverk, moduler, databaseløsninger og lignende programmer vi kunne dra nytte av. Vi fant ut at det eksisterer mange ulike løsninger for å transkribere i sanntid, men ikke like mange for å transkribere opptak.

Søk etter lignende løsninger på markedet viste at det finnes få som er rettet mot å automatisere transkribering av opptak. De fleste hadde kun funksjonalitet som mediespillere med innebygd tekstredigeringsverktøy for å spille av og manuelt transkribere samtidig.

Noe av den mest avanserte og relevante programvaren vi fant var Otter. Slik de beskriver sin programvare, driver de først og fremst med transkribering av møtevirksomhet. Til forskjell fra vår løsning er deres program orientert mot sanntids-transkribering der møter transkriberes underveis. Til å begynne med hadde de integrering mot Zoom, og ble nylig integrert mot Google Meet. Deres programvare selges som et abonnement med månedlig betaling (Otter Voice Meeting Notes, 2021).

2.2.2. Personas og kundereise

Alex
Tech-lead i en startup

33 år
Bor i Oslo

Behov

Har mange møter hver dag, sliter med å få notert fra alle møter og følge med 100%. Trenger et verktøy som gjør dette for seg. Behøver at verktøyet tar i mot lyd og videofiler slik han ikke trenger å konvertere.

Frank
Lærer på videregående

53 år
Bor i Ålesund

Behov

Har en elev med nedsatt hørsel, og vil derf gi h*n mulighet til å lese et transkript av forelesningen, om h*n ikke har fått med se alt. Trenger også et tekstredigerings-verktøy for å formatere teksten.

Linda
Forsker

41 år
Bor i Oslo

Behov

Har mange lange intervjuer. Trenger en måte å raskt finne i tidslinjen hvor ting ble sagt. Vil finskrive intervjuene og plukke ut de viktigste punktene.

Katarina
Elev

13 år
Bor i Trondheim

Behov

Har akkurat flyttet til Norge og har manglende norskkunnskaper. Deltar i digit undervisning på skolen og har opptak. Kan transkribere opptakene og bruke Google Translate til å oversette til sitt språk.

Bente
Pensjonert barnehagelærer

69 år
Bor i Bodø

Behov

Har nedsatt hørsel etter et arbeidsliv i barnehage. Hun er i karantene og hun trenger transkribering av videomøter med legen sin.

Kurt
Student

21 år
Bor i Kristiansand

Behov

Har flyttet alene for å studere. Klarer aldri stå opp til digital forelesning. Utsetter allti å lese til siste liten. Trenger å transkribrere forelesninger slik at han kan skumme

Figur 0: Personas

Personas kan hjelpe til med å adressere noen av problemene med nåværende brukersentrerte tilnærminger. Personas er fiktive og spesifikke representasjoner av brukere. En persona representerer en samling av brukere som deler felles karakteristikker i atferd, som en hypotetisk stereotype av en reell bruker. Anekdotiske bevis fra praksis foreslår at bruken av personas kan legge til rette for nyttig og brukbart design (Miaskiewicz og Kozar, 2011).

Figur 1: Skjermdump av brukerreise.

Som en del av å kartlegge kundereisen i løsningen satte vi opp våre personas i en arbeidsflyt. Der la vi til grunn deres forskjellige roller og behov for hvilke funksjoner de benyttet underveis. Dette ga en oversikt over de forskjellige kundereisene vi har tatt i betraktning under design og utvikling av løsningen.

2.2.3. Brukertester

Testing av brukervennlighet er en stor del av brukerfokusert design. Bruken av prototyper for testing er standard praksis. Prototyper gjør det mulig å teste tidlig design og funksjonalitet når som helst i et utviklingsløp. Sådan er utvikling og testing av prototyper en viktig del av designprosessen. I tillegg til å fungere som modell for en designidé, kan de fungere som en måte å kommunisere mellom brukere, utviklere og designere for å sikre at de deler samme syn (Magnussen, 2010). Brukertester kan også benyttes i innledende fase for å avdekke behov utover resultatene fra de første sprintene. På den måten får man en bedre innsikt i løsningen og brukerne. Det er viktig å stille de riktige spørsmålene når man brukertester (Nielsen, 2012).

3. Prosess og metoder

I dette kapittelet presenteres prosesser og metoder vi har brukt. Vi drøfter hvilke valg vi har tatt basert på egne erfaringer og relevant forskning.

3.1 Utviklingsmetoder

For planlegging, utførelse og administrering av programvareutvikling finnes det forskjellige metoder. Disse kan være vannfall, prototyping, iterative, raske, strukturerte, objekt-orienterte og agile metoder. Alle med sine tilhengere og kritikere. Å velge riktig metode er avgjørende og kan avhenge av flere faktorer. Noen ganger kan det være basert på markedsføring og litteratur-bias som støtter nye eller industri-støttede praktiseringer. Andre ganger vil selskaper stole på egne standarder for struktur og repeterbarhet. Valget vil aldri være enkelt og det er viktig å ta høyde for karakteristikkene til organiseringen, teamet og prosjektet (Vijayasarathy og Butler, 2016, s. 86).

3.1.1 Google Design Sprint

Som en innledende fase la vi opp til å gjennomføre en Google Design Sprint (heretter GDS), for å arbeide med brukergrensesnitt og design. GDS er en arbeidsmetode for rask iterasjon og utvikling av prototyper der man legger vekt på å samle så mange ideer som mulig. Man plukker det beste fra idéene og kombinerer disse til en eller flere prototyper. Vi tok utgangspunkt i boken *Sprint: How to Solve Big Problems and Test New Ideas in Just Five Days,* der vi fulgte stegene den anbefaler gjennom prosessene. Vi vurderte det som en god metode for vårt prosjekt da vi kjente til strukturen, prosessen og resultatene det kunne gi oss. I GDS skal man i utgangspunktet tildele roller. Én som tar avgjørelser, én som er designekspert, én som er finansekspert osv. (Knapp, Zeratsky og Kowitz, 2013, s. 39). Siden vi er tre på gruppen med ganske like forutsetninger, valgte vi å se bort i fra disse større rollene og heller ta avgjørelser sammen. Selv om noen har høyere nivå av design-egenskaper, var alle med på design og prototyping. På den måten kunne den av oss med høyest kompetanse på design komme med viktige tilbakemeldinger, og de to andre kunne forbedre seg og lære mer. De to med mindre designerfaring kunne til gjengjeld komme med andre perspektiver på utformingen.

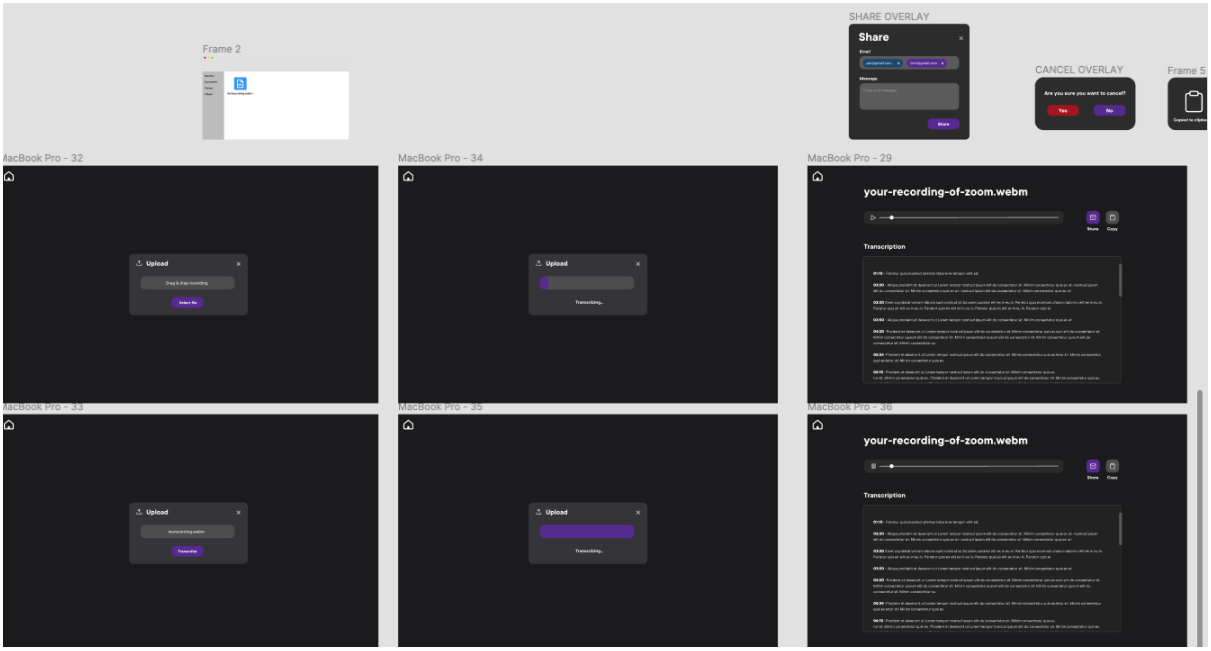
Når man samler ideer og resultater via metodene til GDS vil man i utgangspunktet ha en vegg eller tavle til å samle og visualisere disse på. Siden alle tre har sittet på hvert vårt hjemmekontor, lot dette dessverre ikke seg gjøre. I stedet fikk vi tilgang til Miro av KB. Miro er en nettbasert tavle som alle med tilgang kan bruke samtidig (Se hele Miro-tavlen i vedlegg D: Lenke til Miro).

Det finnes argumenter både for (Alexandros, 2017) og mot (Østerbæk, 2019) GDS. Vi forstår at fem dager i mange tilfeller kan være i korteste laget for å sitte igjen med noe helt konkret. Det har imidlertid ikke vært målet vårt. Målet vårt med GDS har vært å prototype noe med den funksjonaliteten som er viktig og potensiell funksjonalitet som kan være fin å ha i løsningen.

3.1.2. Brukertesting av prototype

Som en del av GDS gjennomførte vi kvalitative brukertester på tre personer. De var kvinner i alderen 25 til 30. På grunn av Covid-19-pandemiens restriksjoner var det utfordrende å få tak i testpersoner.

Testpersonene hadde alle en jobb hvor de deltok i møter og noen av dem skrev referat av møtene sine. Ofte ser man at få testere gir bedre resultat enn en kvantitativ undersøkelse. Dette fordi man raskt kan iterere mellom testene for å få fikset 'feil'. Ved å teste flest mulige versjoner av samme produkt kan må finne frem til den aller beste versjonen (Nielsen 2012).

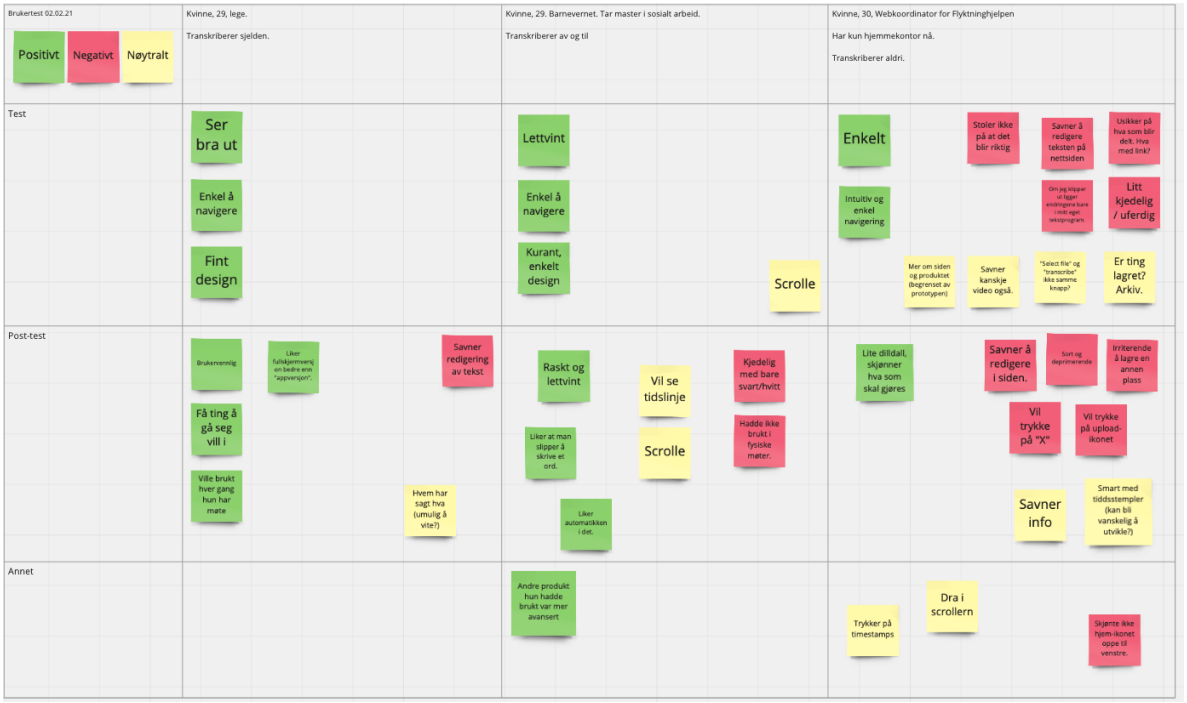


Figur 2: Bilde av prototypen som ble testet

Vi ville se hvordan de interagerte med vår prototype før vi itererte videre. Vi fikk alle testpersonene til å gjennomføre samme test og svare på de samme spørsmålene. Det vi i hovedsak ville finne ut var om løsningen var lett å bruke og om designet var enkelt å forstå. Brukertestene ble holdt digitalt over Discord med skjermdeling. Da kunne vi se hvordan brukeren interagerte med prototypen.

3.1.3. Funn fra brukertester

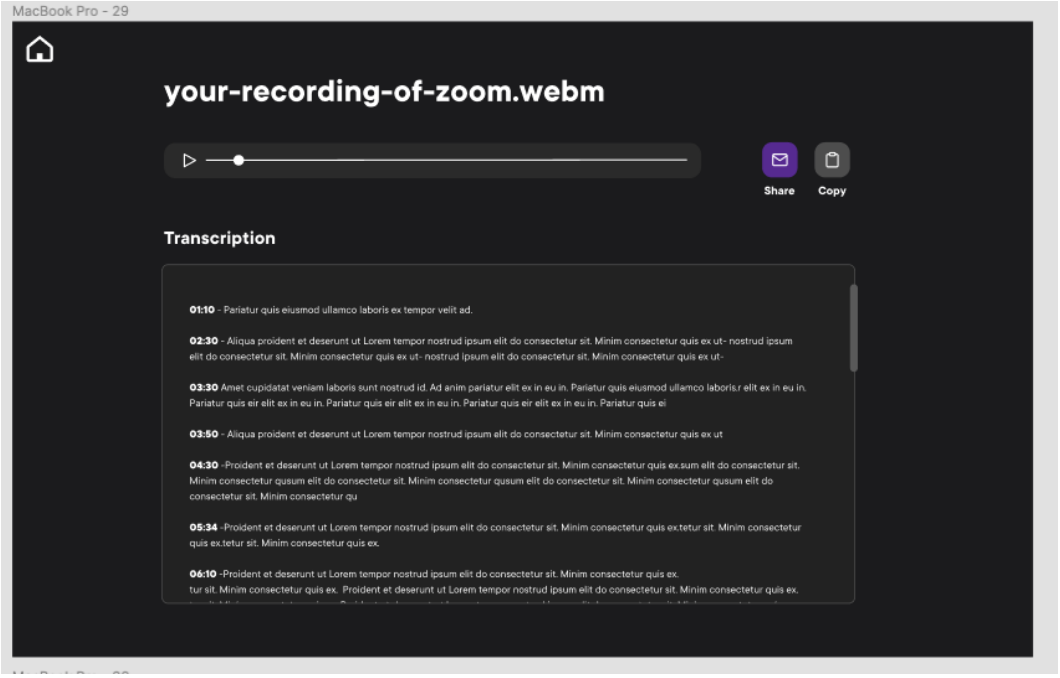
Testpersonene skulle test hovedfunksjonaliteten i løsningen vår. De skulle laste opp et lydklipp og deretter transkribere det. De fikk bare beskjed om hva de skulle gjøre, ikke hvordan de skulle gjøre det. Alle testpersonene ga tilbakemelding om at de fant løsningen enkel å bruke og at det ikke var forvirrende å gjennomføre oppgavene vi ga dem. Det testerne ofte fant forvirrende/irriterende, var funksjonaliteter det ikke er mulig å implementere i Figma. Som at scrollen til tekst-vinduet ikke var flyttbar og at det ikke gikk an å redigere teksten de kopierte.



Figur 3: Tabell over våre funn i Miro. Lenke til Miro:

https://miro.com/app/board/o9J_lZUKoDU=/

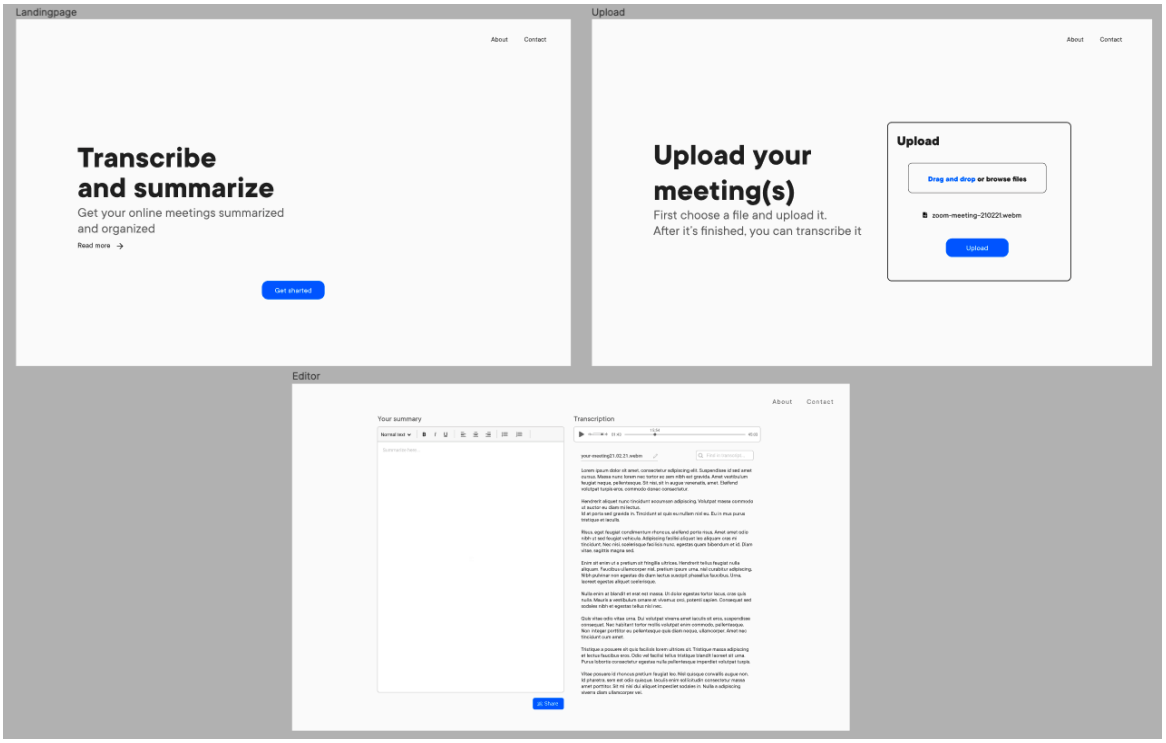
Et annet interessant funn var at en av testerne hadde brukt andre transkriberings-verktøy tidligere som hun beskrev som mye vanskeligere å bruke/navigere. Dette ga oss fin tilbakemelding om at vi var på riktig vei når det gjaldt brukervennlighet.



Figur 4: Tidlig versjon av prototype, lydfil ferdig transkribert uten tekstverktøy.

Designet til prototypen ble beskrevet som “kjedelig og uferdig” og “sort og deprimerende” (Figur 4). Denne tilbakemeldingen fikk vi fra samtlige testpersoner (se i vedlegg E: Brukertester). På bakgrunn av dette bestemte vi oss for å gjøre endringer. Ved neste iterasjon lagde vi en lys variant der vi beholdt det vi fikk positiv tilbakemelding på. Som for eksempel at løsningen var selvforklarende og lett å navigere.

En annen tilbakemelding som skilte seg ut, var at den ene deltakeren i brukertestingen uoppfordret påpekte at når hun transkriberer setter hun av mye tid – gjerne flere timer. Dette var noe vi bemerket oss og tok med videre i utformingen av løsningen.



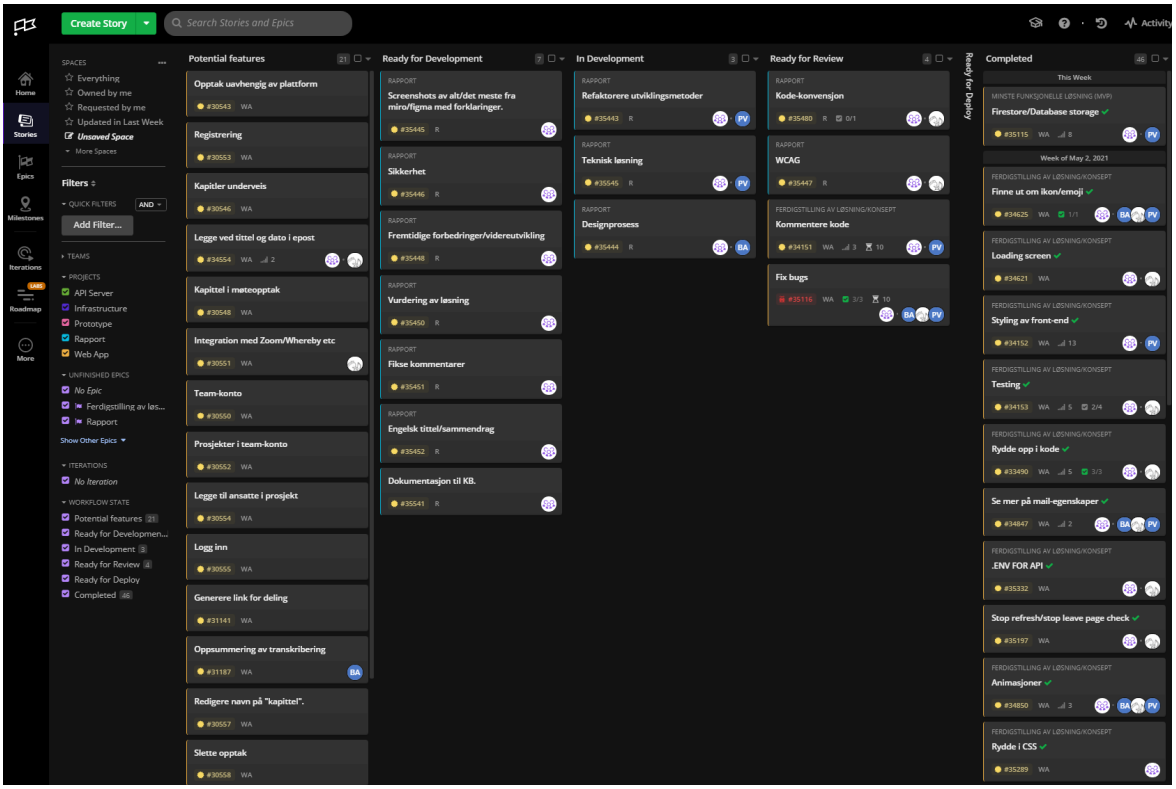
Figur 5: Prototype etter gjennomført brukertesting

Vi viste deretter den itererte prototypen til KB som godkjente den for videre arbeid. Denne prototypen er ganske lik den endelige løsningen.

3.1.4. Scrum

Scrum er en agil, team-basert, produksjonsmodell som beskriver hvordan ett enkelt team kan produsere verdifulle resultater (ofte kalt produkt) for interessenter i et komplisert utviklingsmiljø (Rawsthorne og Shimp, 2018, s. 4). For å lese mer om Scrum, se vedlegg F: Om Scrum. Helt fra start har vi arbeidet på denne måten. Siden oppgavene har vært varierte, har også varighet på sprintene variert. I et prosjekt hvor en gjør alt fra å designe til å kode, vil det være hensiktsmessig å dele det opp i mindre sprinter. Måten vi har delt det opp på har vært basert på den opprinnelige fremdriftsplanen, med justeringer ettersom oppgavene ble gjennomført. Vi tok erfaringene vi har fra denne typen prosjektarbeid med oss når vi skisserte opp fremdriftsplanen.

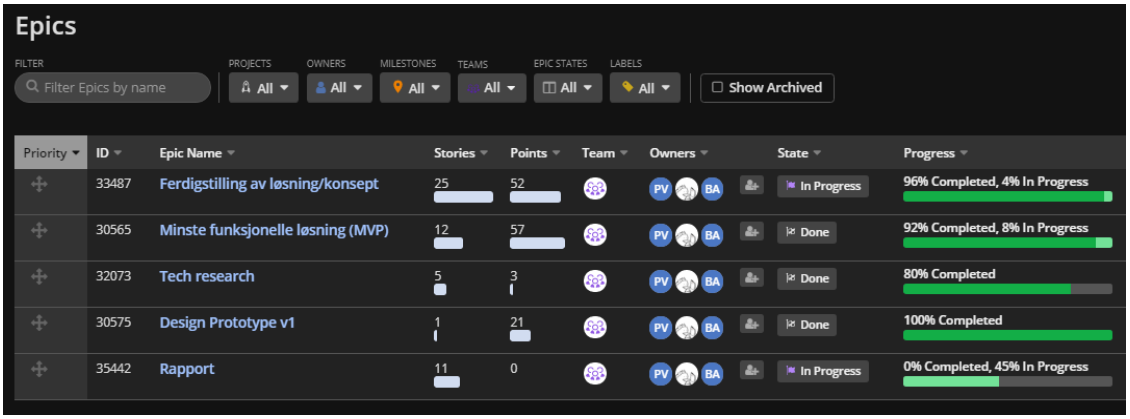
3.1.5. Kanban



Figur 6: Skjermdump fra Clubhouse Stories mot slutten av prosjektperioden.

Kanban brukes for å visualisere arbeidsoppgaver på en tavle. Oppgavene og deres detaljer skrives på kort, og festes til tavlen. En Kanban-tavle deles opp i kolonner som tilsvarer arbeid som skal gjøres, arbeid som gjøres nå og arbeid som er ferdig. Les mer om Kanban i vedlegg G: Om Kanban.

KB ga oss tilgang til prosjektstyringsverktøyet Clubhouse, som lot oss lage kort på en digital tavle. Her kunne vi lage såkalte “epics” for de mer overordnede stadiene av prosjektperioden.



Figur 7: Skjermdump fra Clubhouse Epics mot slutten av prosjektperioden

For hver Epic lagde vi kort ("stories" i Clubhouse) for hver arbeidsoppgave eller funksjon. Vi brukte fem kategorier på tavlen vår: Potensiell funksjonalitet, klar for utvikling, under utvikling, klar for anmelding og ferdig. Kolonnen for "potensiell funksjonalitet" brukte vi for å legge inn alle ideer for funksjonalitet. Disse diskuterte vi for og imot med hensyn til tid, arbeidsmengde og nytte for MVP-en før vi flyttet de funksjonene vi mente var viktigst til "klar for utvikling". De som ikke ble valgt, lot vi stå for eventuell fremtidig utvikling av løsningen. Ved å ta i bruk epics og stories hadde vi hele tiden kontroll på arbeidsoppgavene og deres status.

3.1.6. Scrum og Kanban sammen

På bakgrunn av våre erfaringer med Scrum og Kanban fra tidligere prosjekter var vi komfortable med å ta i bruk disse i vår prosess. Vi kunne dermed trekke ut de aspektene ved begge som vi likte. Ut ifra dette ble vår utviklingsmetode etter GDS en hybrid av disse to. Med tanke på at gruppen vår består av tre medlemmer, ble vi enige om at vi ikke skulle angi roller som Product Owner eller Scrum Master. Vi kom til at avgjørelser skulle tas sammen og at alle skulle hjelpe hverandre med å holde gruppen oppdatert på prosessen. Likevel har vi sett på KB som vår Product Owner og involvert de i prosessen med å finne ut av hvilken funksjonalitet som har vært viktig å ha med i løsningen. Fra Scrum har vi i hovedsak tatt i bruk sprinter, iterasjoner og Daily Standup. De første sprintene besto av GDS før vi gikk over til en mer research-basert sprint for å finne de riktige verktøyene vi kunne bruke i utviklingen. Vi sammenlignet verktøyene og plukket ut de vi syntes var riktige for oss. Deretter videreførte vi disse til en utviklings-sprint med fokus på funksjonalitet fremfor

design. Til slutt hadde vi en sprint med fokus på design og brukervennlighet. Vi tok i bruk prototypene våre fra GDS og baserte designet i stor grad på disse. Da det hadde gått noe tid siden vi sist hadde tenkt på design, hadde vi opparbeidet oss noen nye meninger om hvordan vi ønsket at løsningen skulle se ut. Den ble i basert på vårt endelige prototype-design, men med noen av våre nye ideer implementert.

Før hver sprint satte vi av tid til å finne ut av hva vi skulle gjøre. Til dette anvendte vi Kanban. Kortene i hver epic fikk blant annet en poengtildeling. Siden målet ikke var å være veldig presise i poengdelingen, lot vi være å bruke en standard tallrekke, men heller en Fibonacci-sekvens (0, 1, 2, 3, 5, 8, 13, 21, 34) (Rubin, 2013, s. 7). For oss representerte de lavere tallene funksjoner som ikke var så viktige der og da, mens de høyere tallene indikerte viktige funksjoner som gjerne kunne ta lenger tid. Vi valgte deretter oppgaver basert på hvilken sprint vi var i og kunne hele tiden holde hverandre oppdatert. Siden vi gjorde mye parprogrammering, var det enkelt å holde følge med hvem som hadde gjort hva.

For å oppsummere Scrum-forløpet vårt med sprinter så det slik ut:
Sprint 1: Google Design Sprint. Første plan basert på mobilapplikasjon med kapittel.
Sprint 2: Google Design Sprint. Oppdatert plan med nytt fokus på et nettbasert verktøy.
Sprint 3: Research. Finne verktøy og teknologier som ville passe i løsningen. Lære oss å bruke disse.
Sprint 4: Utvikling. Koding og bruk av verktøy for å bygge løsningen.
Sprint 5: Stilsetting og testing av løsningen.

3.2. Fremdriftsplan

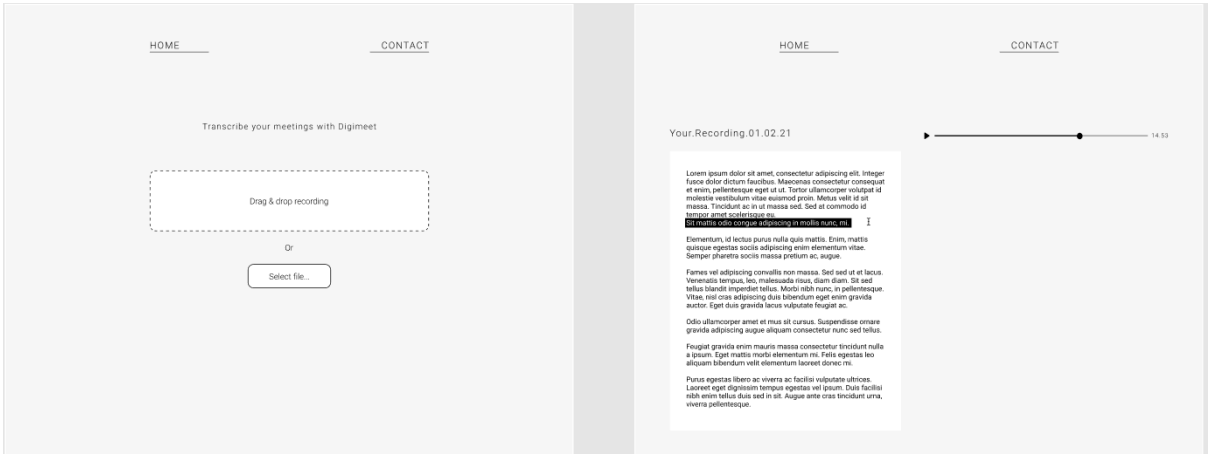
I starten av prosjektperioden la vi opp en grov fremdriftsplan for å holde orden på frister og tiden vi hadde til rådighet. På bakgrunn av innspill fra KB ønsket vi å legge utviklingsløpet i to sykluser. Ved å iterere flere ganger ville det trolig styrke løsningen. Grunnet kompleksiteten og tiden det ville ta å sette seg inn i verktøyene, innså vi etter et par uker at det ikke lot seg gjøre. Vi bestemte oss dermed for å legge opp et nytt løp med færre, lengre sprinter som ga oss bedre tid til å utvikle en god første iterasjon av løsningen (se vedlegg H: Fremdriftsplaner).

3.3. Parprogrammering

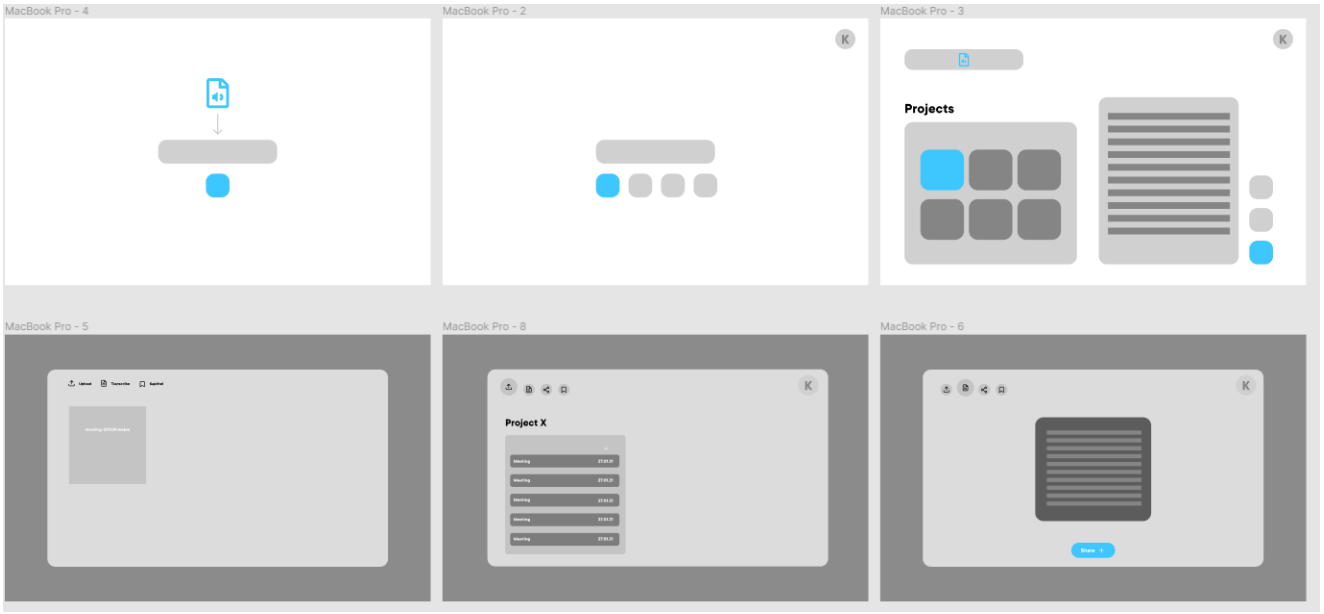
“Pairs find that by pooling their joint knowledge, they can conquer almost any technical task immediately” (Williams, Kessler, Cunningham og Jeffries, 2000). Parprogrammering legger et press på deltakerne for å prestere og holde hverandre fokuserte. På grunn av dette presset oppleves det gjerne en større motivasjon for å fullføre oppgavene i løpet av økten. Dermed arbeider man med større fokus og intensitet enn om man arbeider individuelt. Nesten alle de kartlagte profesjonelle programmererne sa at de hadde større tillit til løsningene sine når de var parprogrammerte (Williams, Kessler, Cunningham og Jeffries, 2000). Vi tok i bruk metoden parprogrammering når vi utviklet vår løsning. Dette er en teknikk vi lærte i løpet av utdanningen, hvor man bytter på å programmere og observere/gi tilbakemelding. Vi lærte mye av hverandre gjennom denne teknikken da den ga et godt innblikk i hvordan andre på gruppen arbeider. Vi byttet på å dele skjerm og på å observere. På grunn av kompleksiteten til mange av funksjonene i løsningen, gjorde parprogrammering og skjermdeling at vi enklere kunne samarbeide og løse oppgavene mer effektivt. Flere personer på samme oppgave har vært viktig både for effektivitet og læring. Dette fungerte også godt på mindre komplekse problemer, som når vi for eksempel arbeidet sammen med designet av løsningen i CSS.

3.4. Design og prototyping

Etter endt GDS hadde vi en rekke utkast av prototyper til løsningen som kunne være aktuelle. For å utfordre oss selv, og samtidig raskere eliminere de aspektene som ikke skulle bli med i den endelige løsningen, bestemte vi oss for å lage hver vår prototype på design i Figma.



Figur 8: Skjermdump tidlig prototype

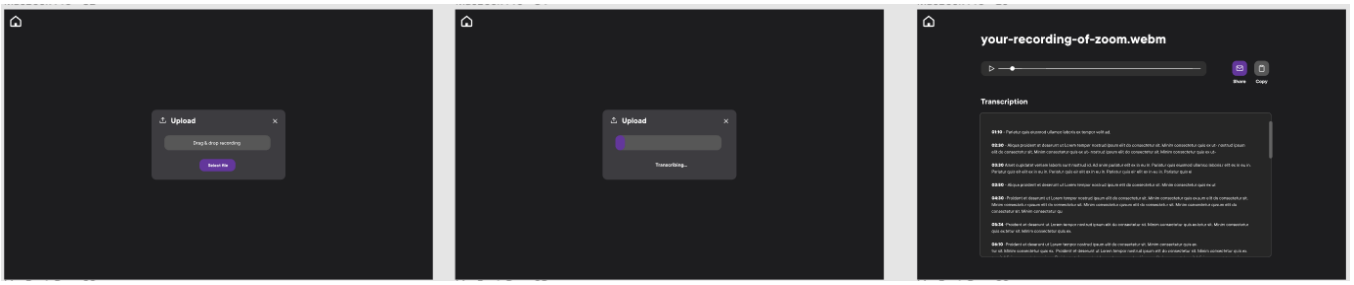


Figur 9: Skjermdump tidlig prototype

Noen av de første prototypene inkluderte også et system for å lage konto for en bedrift/skole for å holde orden på alle opptakene. Vi erfarte ganske tidlig at dette ville ta for lang tid å implementere i tillegg til hovedfunksjonalitet, derfor ble det utelatt.

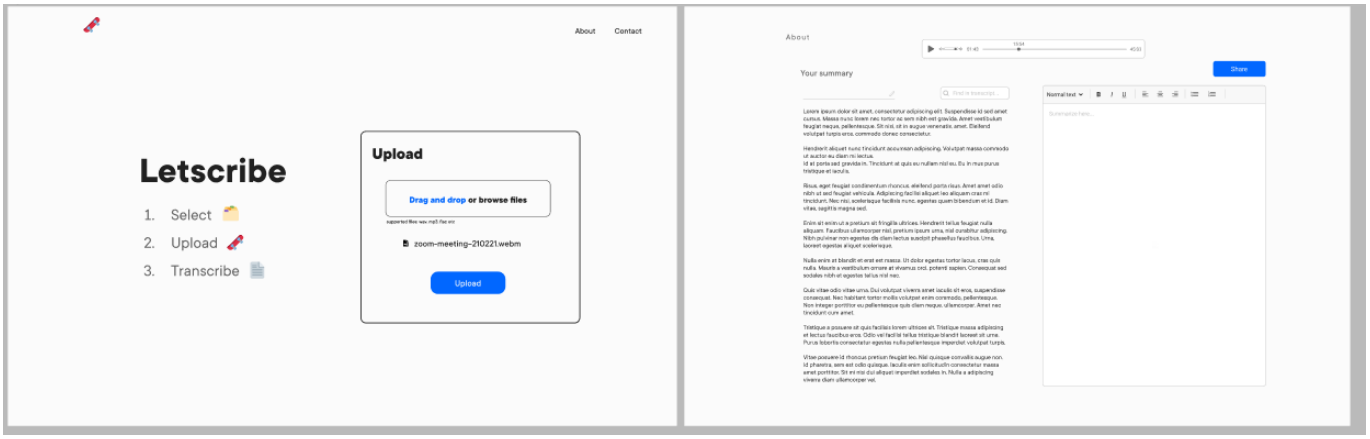
Figur 10: Skjermdump tidlig prototype

Figur 10 viser noe av det som vi fortsatt brukte som inspirasjon fra den første ideen om en mobilapplikasjon, deriblant mediespilleren og tidsstempler. Der kunne man også sette "kapitler" for å lettere finne de viktige delene i opptaket i etterkant.



Figur 11: Skjermdump tidlig prototype

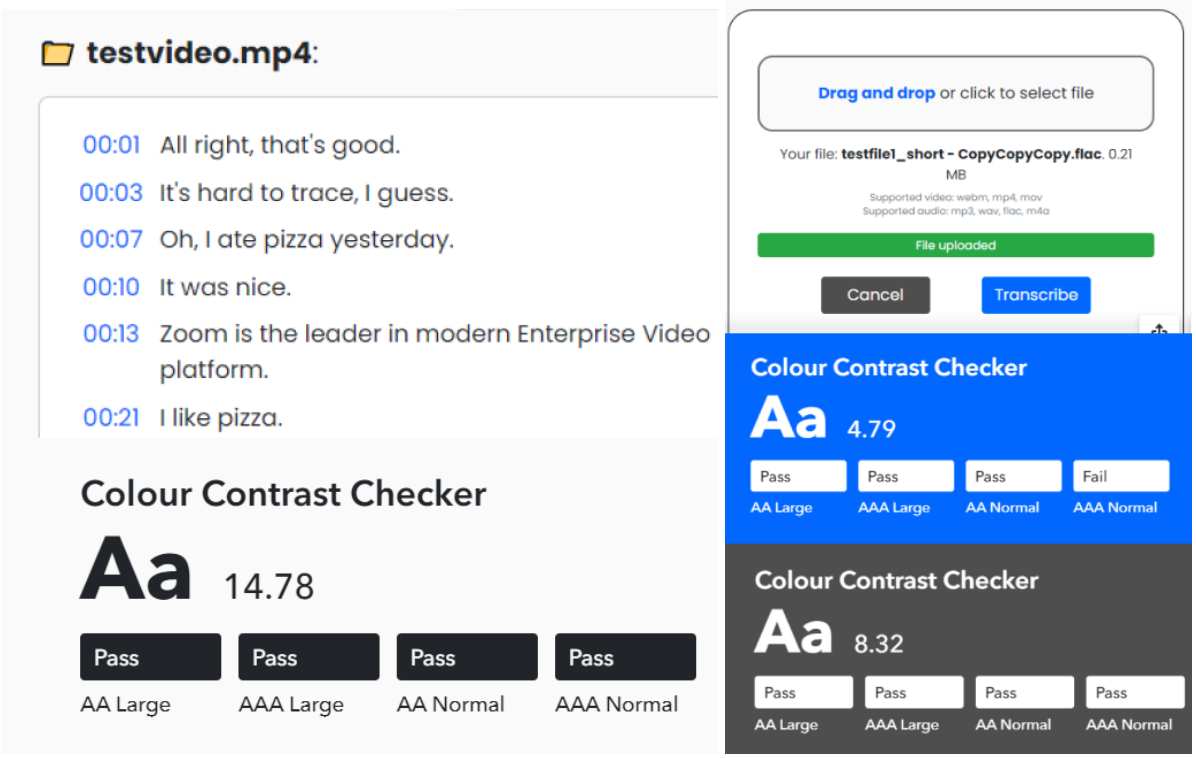
Etter de første versjonene valgte vi ut det vi mente var de beste aspektene fra alles prototyper, før vi satte sammen det vi alle var fornøyd med. Vi itererte noen ganger på prototypene til vi hadde et resultat som både vi og KB syntes var tilfredsstillende. Fokuset på enkelhet og brukervennlighet var viktig, deriblant bruken av farger. Ved bruk av bare én aksentfarge og resten i svart og hvitt er det lett å styre fokuset til brukeren der vi vil (Using Color to Confuse and to Prevent Unwanted Actions, 2020).



Figur 12: Skjermbilde av endelig prototype med tekstverktøy.

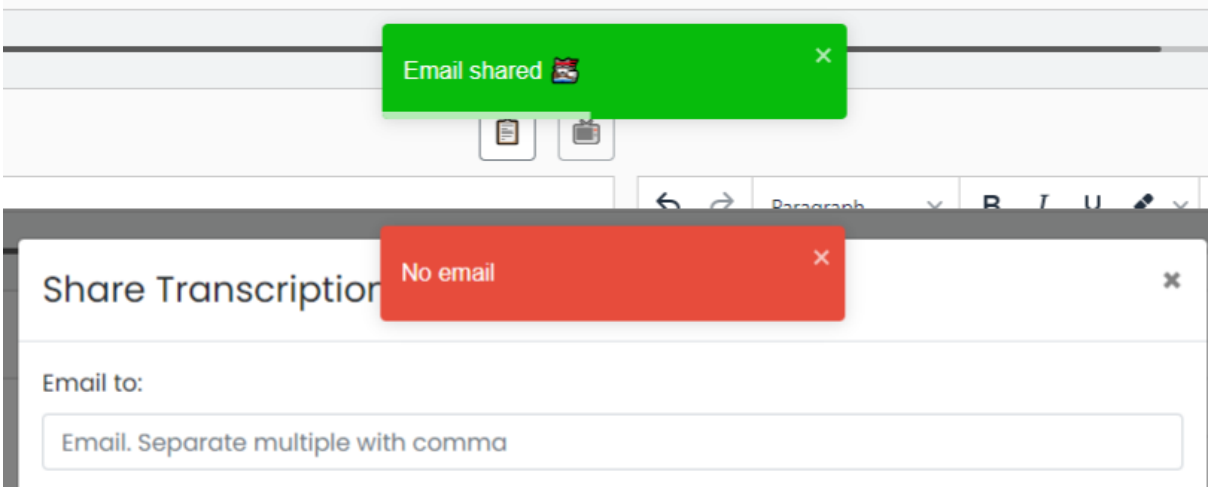
3.4.1. Universell utforming

Vi har i vårt arbeid fulgt WCAG 2.0-modellen (Web Content Accessibility Guidelines, 2021). Den definerer hvordan man gjør nettinnhold mer tilgjengelig for personer med nedsatt funksjonsevne. Løsningen vår skal kunne benyttes av alle, derfor har vi tatt følgende steg for å sikre dette: Alt-tags (Alternativ tekst-attributt) i HTML-koden sørger for at brukeren får mulighet til å navigere med tastaturet, og i tillegg kan de med skjermleser få tilbakemelding via lyd. For fargekontrast er det 3 nivåer: A, AA og AAA, der AAA er best. Dersom en kontrast-sjekk er bestått på alle nivåene har løsningen en god kontrast og er lesbart. Selv om AAA er det best oppnåelige er det ikke alltid nødvendig da det er vanskelig å oppnå for alt innhold i en nettside (Web Content Accessibility Guidelines, 2021).



Figur 13: WCAG-test for lesbarhet

God lesbarhet er viktig for vår løsning. Hvit bakgrunn med svart tekst gir trolig den beste lesbarheten. Lesbarhet på knappene er også godt innenfor kravene til universell utforming/WCAG, samtidig som aksentfargen gir god indikasjon på at dette elementet er noe man kan interagere med.

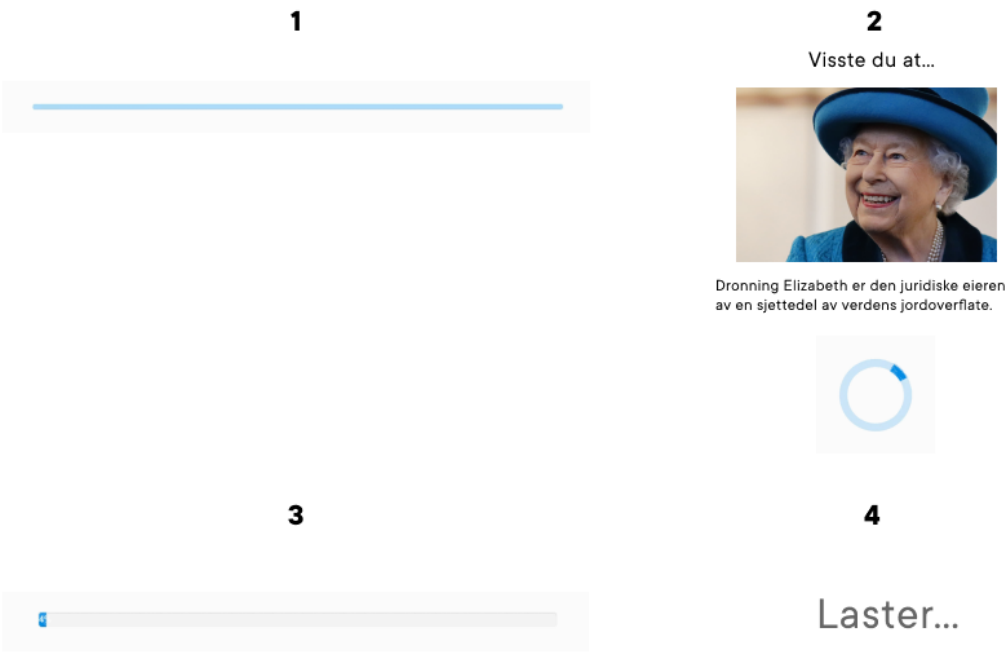


Figur 14: Tilbakemeldinger

Tydelige tilbakemeldinger når brukeren utfører en handling sikrer at de får med seg at noe har skjedd og gir de muligheten til å rette eventuelle feil. Det hindrer også forvirring om for eksempel en e-post er sendt eller ikke. Ved hjelp av disse tilbakemeldingene kan brukeren lettere forstå det som skjer i løsningen. Løsningen kan også skaleres opp til 400% og fungere som normalt, men med noe redusert brukervennlighet.

3.5. Brukerundersøkelse av laste-elementer

Som tema for vår eksamensoppgave i faget Undersøkelsesmetoder, undersøkte vi om forskjellige måter programvare viser at den laster kan påvirke hvordan brukerne oppfatter ventetid. Vi oppdaget underveis at vi hadde to steg i programvaren der brukerne må vente på at noe skal laste. Først ved opplasting av fil, deretter ved transkribering av filen. Prototypene vi laget tok høyde for lasting ved transkriberingen, men ikke ved filopplasting. Hvor lang tid brukeren måtte vente var da heller ikke vurdert. Dette ga oss en utfordring siden ventetiden var forskjellig for de to stegene, lengst ved transkriberingen. Man kan da vurdere om det var riktig å bruke den samme laste-animasjonen for begge stegene. Ventetiden kunne også variere ut ifra hvor stor fil vi testet med. Et spørsmål er om brukeren burde få en indikasjon på hvor lang tid som gjenstår.



Figur 15: Laste-animasjonene vi testet

Etter å ha gjort litteratursøk på temaet utformet vi en undersøkelse for å teste vår teori. Brukerne ble bedt om å se på bilder av dyr. Mellom hvert dyr ble de presentert med en laste-animasjon som varte i 12 sekunder (Figur 15). Det ble aldri nevnt hva målet med testen var, slik at brukerne ikke skulle underbevisst ta tiden til hver animasjon.

Alle brukerne rapporterte at de følte at laste-animasjonene tok forskjellig tid, selv om alle varte i 12 sekunder. Resultatet var at vi fikk størst tilfredshet hos brukerne ved å benytte laste-elementer som viste fremgang eller resterende tid. Vi skulle aller helst brukt denne typen laste-animasjon ved begge stegene i løsningen, men det viste seg å ikke være mulig å hente nødvendig informasjon fra Google sitt API. Derfor ble en spinner, som brukerne også var tilfreds med, benyttet ved transkriberingen mens statusbar og prosent ble benyttet ved opplasting.

3.6. Verktøy

Vi brukte flere verktøy til prosjektstyring, organisering, kommunikasjon, prototyping og utvikling. I denne delen forteller vi kort om verktøyene vi anvendte og hvordan de ble brukt.

3.6.1. Figma

Figma er et prototypingsverktøy med høyt fokus på samarbeid. Her kan hele teamet arbeide samtidig og iterere på de samme prototypene. Da hjemmearbeid har vært aktuelt under hele perioden, har Figma vært et verdifullt verktøy for samarbeid.

3.6.2. Miro

Miro er en en nettbasert tavle for å samarbeide på en visuell måte. Det ga oss mulighet til å fullføre GDS på omtrent samme måte som det ville blitt gjort om vi var fysisk til stede i samme rom. Miro ga oss også muligheten til å gjennomføre mange av de samme teknikkene brukt i GDS via forskjellige utvidelser til verktøyet. Som for eksempel å gi poeng via “dotter” i utvelgelsesprosessen. Dotter fungerer som en snarvei for å unngå lange diskusjoner og

samtidig finne ut hva man bør prioritere (Knapp, Zeratsky og Kowitz, 2013, s. 87). Vi prøvde forskjellige gratisprogrammer tilsvarende Miro, men ingen var like enkle eller hadde funksjonaliteten vi trengte.

3.6.3. Clubhouse

For å prioritere funksjonaliteter og mål med utviklingsperioden brukte vi Clubhouse. Clubhouse gir mulighet til å holde styr på alle mål for funksjonalitet. Man kan vurdere hvor ressurskrevende en oppgave vil være å komme i mål med. Det er et godt verktøy å bruke for å planlegge ressursbruk når det kommer til oppdragsgivers prioriteringer og mål. Clubhouse er ikke gratis, men vi fikk tilgang via KB. Dermed kunne de også følge med på utviklingen.

3.6.4. Discord

Vi brukte Discord som kommunikasjonsplattform. På Discord kan man lage kanaler for både tekst og tale. Ved å kunne dele skjerm var det enklere å samarbeide, særlig i forbindelse med parprogrammering. Parprogrammering kom til å bli en stor del av hvordan vi samarbeidet og siden vi hadde hjemmekontor måtte vi bruke skjermdeling. Discord har et valg for å gjøre tekst klarere, slik at kode blir lettere å lese for de som observerer.

3.6.5. Visual Studio Code

Integrated development environments (IDE-er), som Visual Studio tilbyr verktøy som automatiserer oppgaver som refaktorering, automatisk fullføring og retting av kompileringsfeil. Målet til disse verktøyene er å øke utviklingshastigheten og minske feil (Muşlu et al., 2012). Vi har brukt Visual Studio Code (VSCode) for tekstredigering av kode, kompilator for oversetting og kjøring av kode, og feilsøkingsverktøy (debugger). Det har hjulpet oss skrive kode raskere og bedre.

3.7. Kode

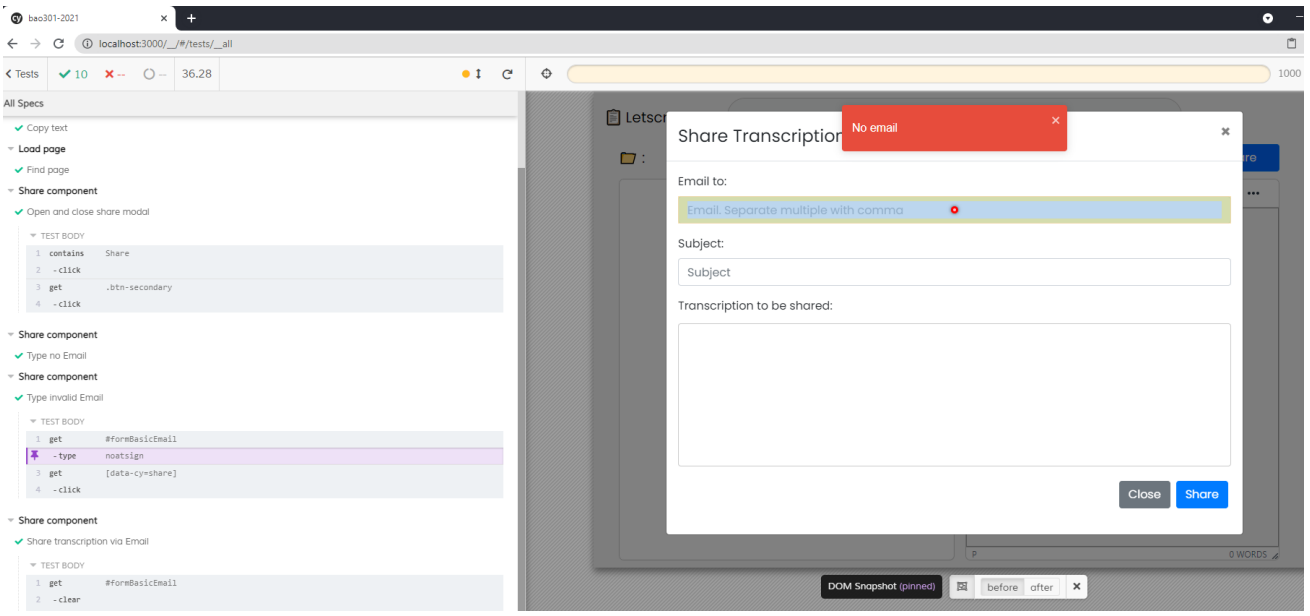
I denne delen av rapporten går vi gjennom valgene vi har tatt for programmeringsspråk, kodekonvensjoner og versjonskontroll.

3.7.1. Programmeringsspråk

Løsningen er kodet i JavaScript (JS). Frontend, den delen av løsningen en bruker ser og interagerer med, er skrevet i ReactJS som er et JavaScript-bibliotek som bygger videre på JavaScript. Backend, det som skjer i bakgrunnen som brukeren ikke ser, er skrevet i JavaScript ved hjelp av Node.JS som webserver-plattform. I React kan man skrive erklærende og komponentbasert som gjør det mer forutsigbart og lettere å finne feil. Erklærende programmering betyr at man forteller programmet hva som skal gjøres, fremfor hvordan det gjøres (Pereiro, 2021). Hvordan er opp til kompilatoren. Ved å dele ting opp i mindre komponenter har man bedre oversikt over hvilken kode som gjør hva (React – A JavaScript library for building user interfaces, 2021). Tester er skrevet med hjelp av Cypress.

3.7.2.Testing av kode

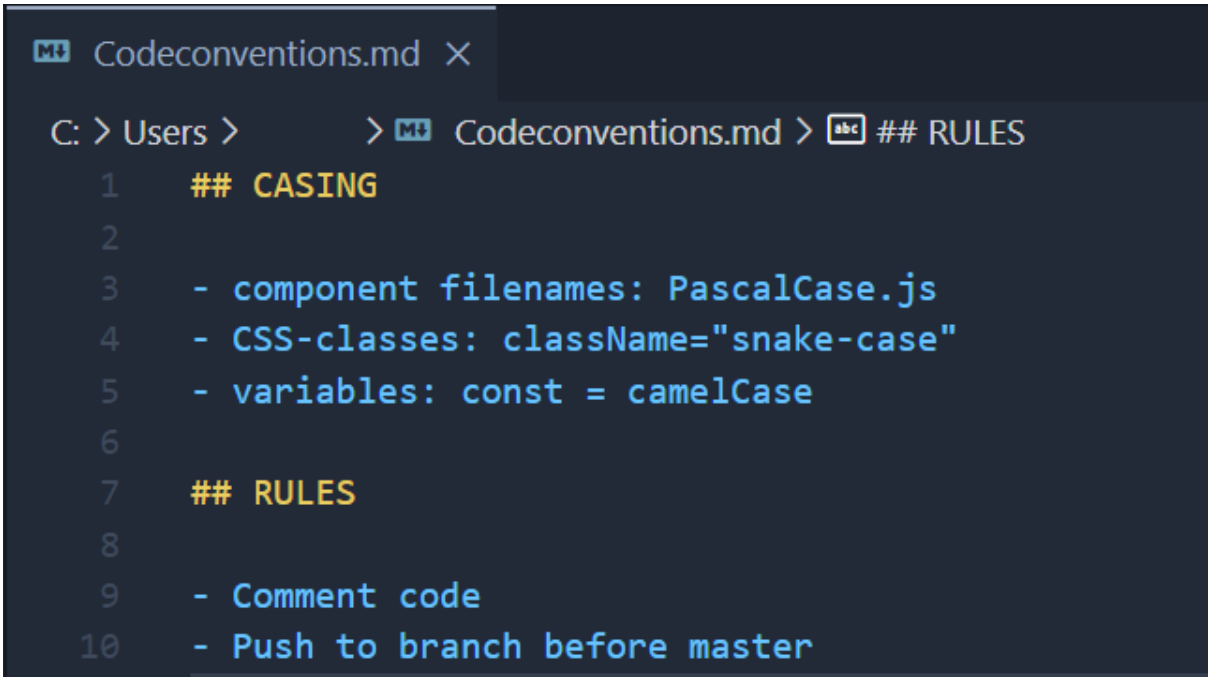
Cypress er et verktøy for å kjøre tester i koden for å forsikre seg om at alt fungerer som det skal før produksjon. Det som skiller Cypress fra andre testverktøy er at man kan se testene kjøre i frontend i sanntid. På denne måten får man oversikt over hva som fungerer eller ikke. I motsetning til andre måter å teste på, kan man trolig spare tid ved at Cypress er enkelt å forstå på grunn av den gode dokumentasjonen og de visuelle tilbakemeldingene.



Figur 16: Skjermdump av Cypress som kjører tester.

3.7.3. Kodekonvensjoner

Vi satte noen faste kodekonvensjoner da vi startet å kode. Dette for å unngå at det ble tre forskjellige måter å skrive kode på, men også for å ha en helhetlig flyt i koden. Slik unngår man for eksempel at variabler blir skrevet på ulike måter. Ved å gjøre dette unngikk vi også forvirring og potensielle feil i koden basert på feil navngivning. Vi ble enig muntlig før vi skrev de ned i sin egen kodekonvensjon-fil i prosjektet. Vi inkluderte også noen regler for bedre flyt i utviklingen; kommentere kode der det var nødvendig, og å dytte ("pushe") de nyeste oppdateringene til egen gren ("branch") i Github før man samlet ting i master. Dette for å sikre at vi alltid hadde en fungerende løsning på master vi kunne gå tilbake til.



Figur 17: Skjermdump av våre kodekonvensjoner i prosjektet

3.7.4. Versjonskontroll

Versjonskontroll lar utviklere beholde historikk av kildekode og prosjektfiler. Det lagrer informasjon for hver fil (og prosjektets struktur) i det som kalles et oppbevaringssted ("repository"). Innenfor et repository kan man ha flere grener. Grener er nyttige når man vil beholde én gren for vedlikehold av en stabil versjon av prosjektet (heter som regel "master branch"), mens andre versjoner under utvikling kan arbeides med i andre grener (Ruparelia, 2010).

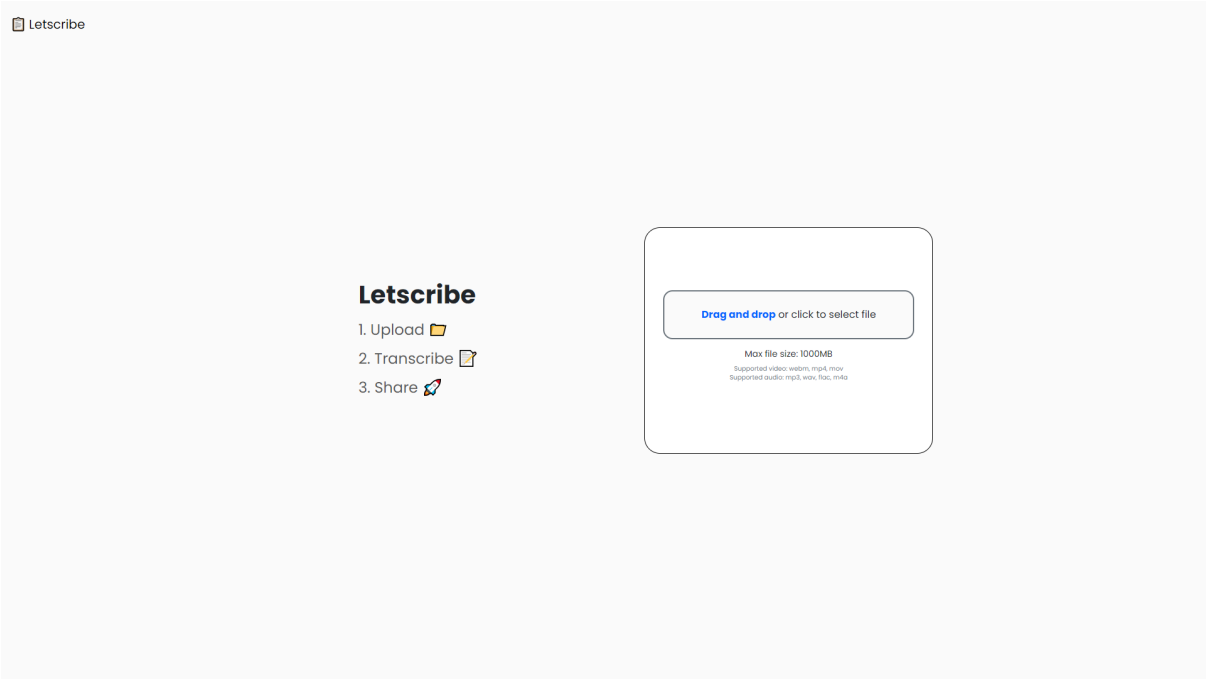
Vi brukte Git for versjonskontroll og kildekode-håndtering (Git, 2021). For å ha bedre oversikt over versjoner og grener brukte vi Github sammen med deres verktøy Github Desktop. Github er en plattform for lagring og oppbevaring av kode som lar utviklere arbeide sammen hvor som helst med prosjekter (Hello World · GitHub Guides, 2021).

3.8. Morgenrituale

Hver morgen før Daily Standup spilte vi Gartic. Et tegne-og gjettespill for å sette i gang hjernen og lette på humøret før arbeidsdagen. “A key function of games at work has been to provide respite and recovery and, in doing so, improve the positive affect people feel when they are at work” (Mollick og Rothbard, 2013). Bruken av spill på arbeidsplassen har vært notert av organisasjonsforskere siden 1930-tallet. Det finnes også dokumentasjon på at bruken av spill på arbeidsplassen kan ha oppstått mye tidligere og ble brukt til å motivere arbeidere så langt tilbake som det gamle Egypt. Man kan argumentere for at å alltid spille et spill hver dag kan ses på som “mandatory fun”. Dersom overordnet velger et spill som ansatte skal spille, kan det føles påtvunget (Mollick og Rothbard, 2013). Heldigvis for oss har vi hatt friheten til å velge dette selv. Gartic har vært en god blanding av konkurranse og kreativitet der det alltid er variasjon, noe som har gjort at det ikke føles repeterende å gjøre hver dag.

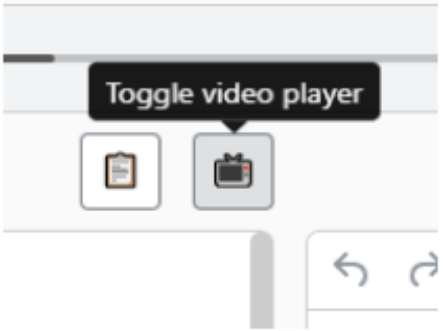
4. Løsning

Vår løsning er et nettbasert verktøy som transkriberer lyd- og videofiler til tekst. Målet med løsningen var å lage et produkt som potensielt kunne erstatte notering i møter.

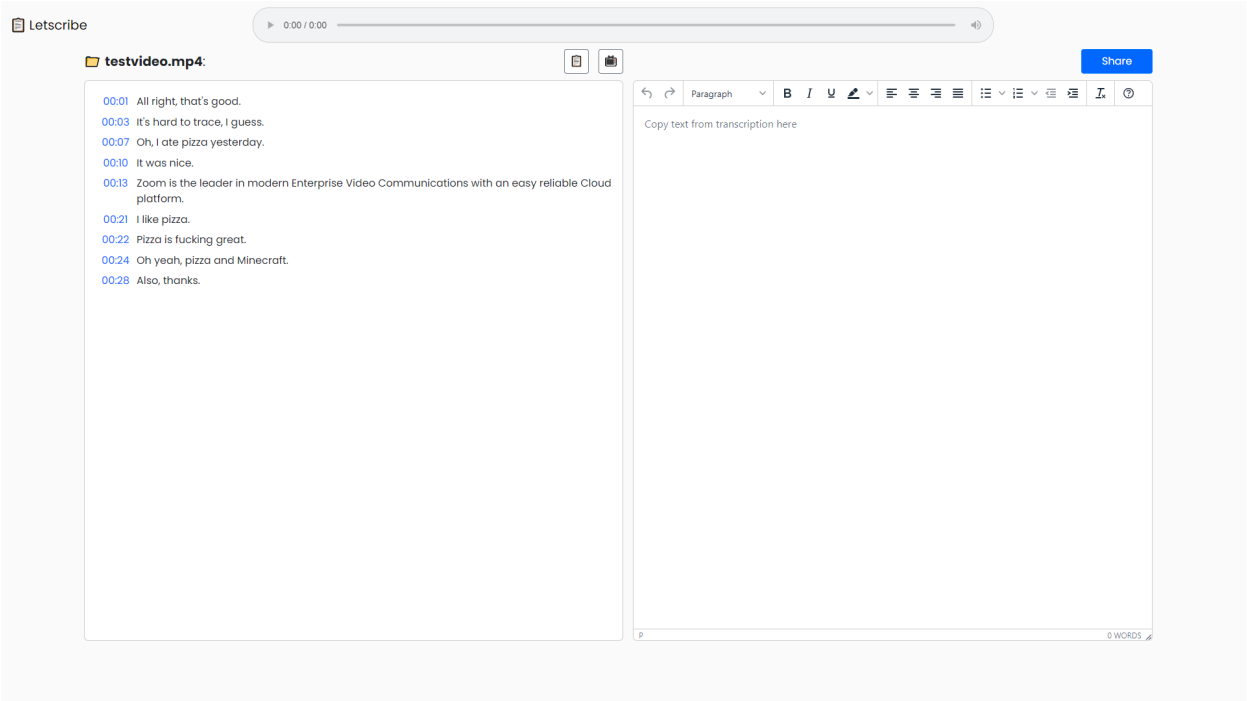


Figur 18: Skjermdump fra fremsiden til vår løsning

På forsiden blir man møtt med en stegvis forklaring på hvordan løsningen fungerer – dette gjør det enklere for nye brukere. Løsningen lar brukeren laste opp en lyd- eller videofil som verktøyet transkriberer. Deretter kan man skrive eget sammendrag av møtet i tekstredigeringsverktøyet. Det er også mulighet for å høre opptaket samtidig. Hver setning i den transkriberte teksten har et tidsstempel som man kan trykke på for å hoppe til tiden i opptaket. Dersom man tror at transkriberingen kan være feil et sted, kan man lett dobbeltsjekke dette ved å høre på opptaket der man er usikker. Man kan også trykke på en knapp for å veksle mellom lyd- eller videospiller, avhengig av hva slags type fil man har lastet opp.

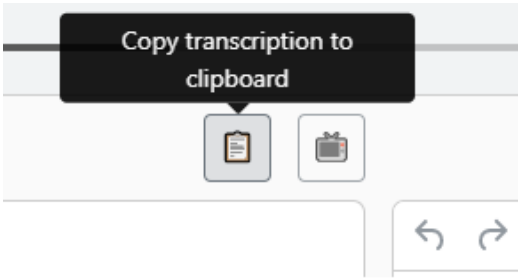


Figur 19: Tidsstempel basert på punktum.

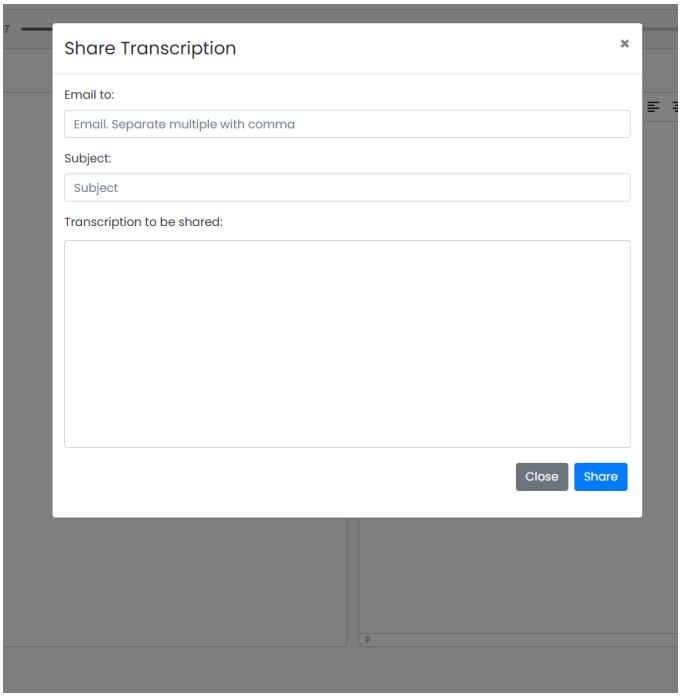


Figur 20: Skjermdump av ferdig transkribert fil.

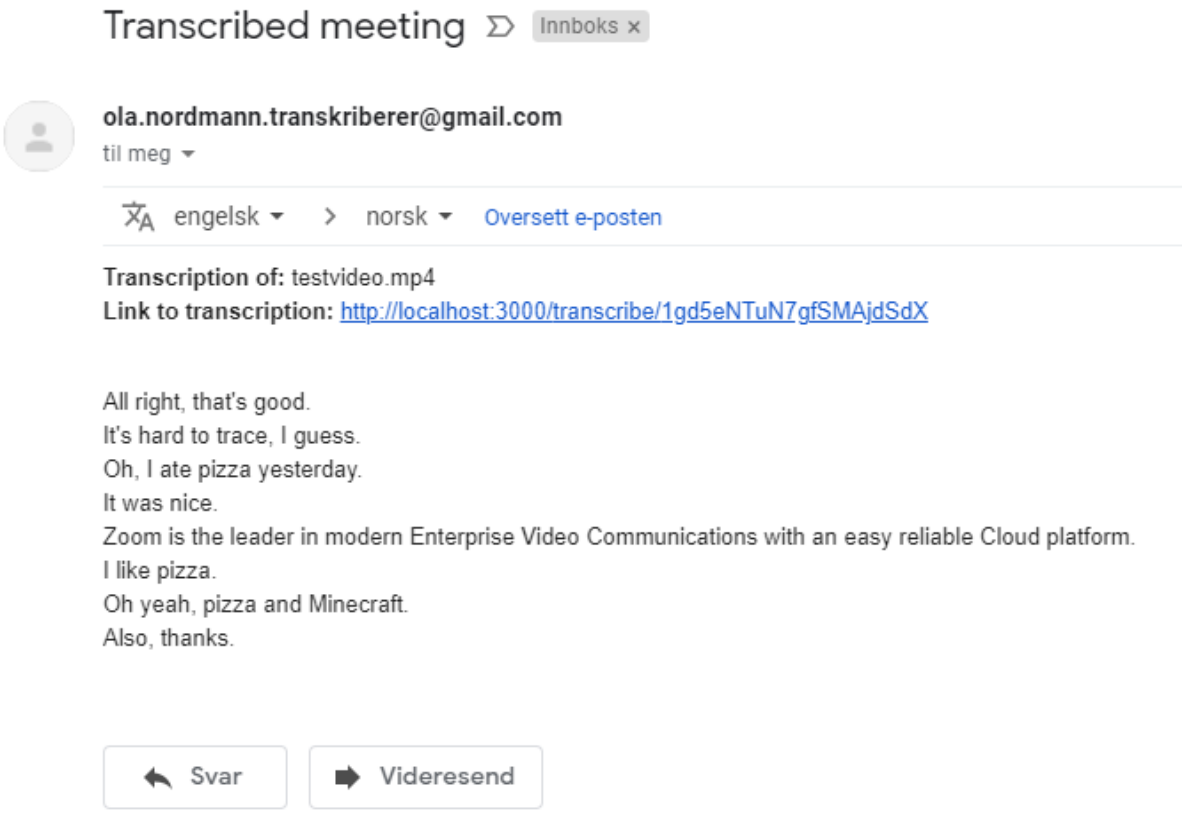
I venstre kolonne er den transkriberte teksten delt inn i setninger. Hver setning har fått sitt eget tidsstempel som spiller av lyden/videoen fra det tidspunktet. I den høyre kolonnen finner vi et tekstredigeringsverktøy med de fleste vanlige funksjoner. Den vanligste bruken av løsningen vil være å kopiere transkriberingen over til tekstverktøyet via kopier-knappen og deretter rette eventuelle feil som kan ha oppstått. Videre kan man dele sin oppsummering til de andre deltakerne ved hjelp av e-post. I e-posten får man oppsummering, samt en lenke tilbake til siden hvor man kan høre opptaket og se den originale transkriberingen.



Figur 21: Tidsstempel basert på punktum.



Figur 22: Skjermdump av deling

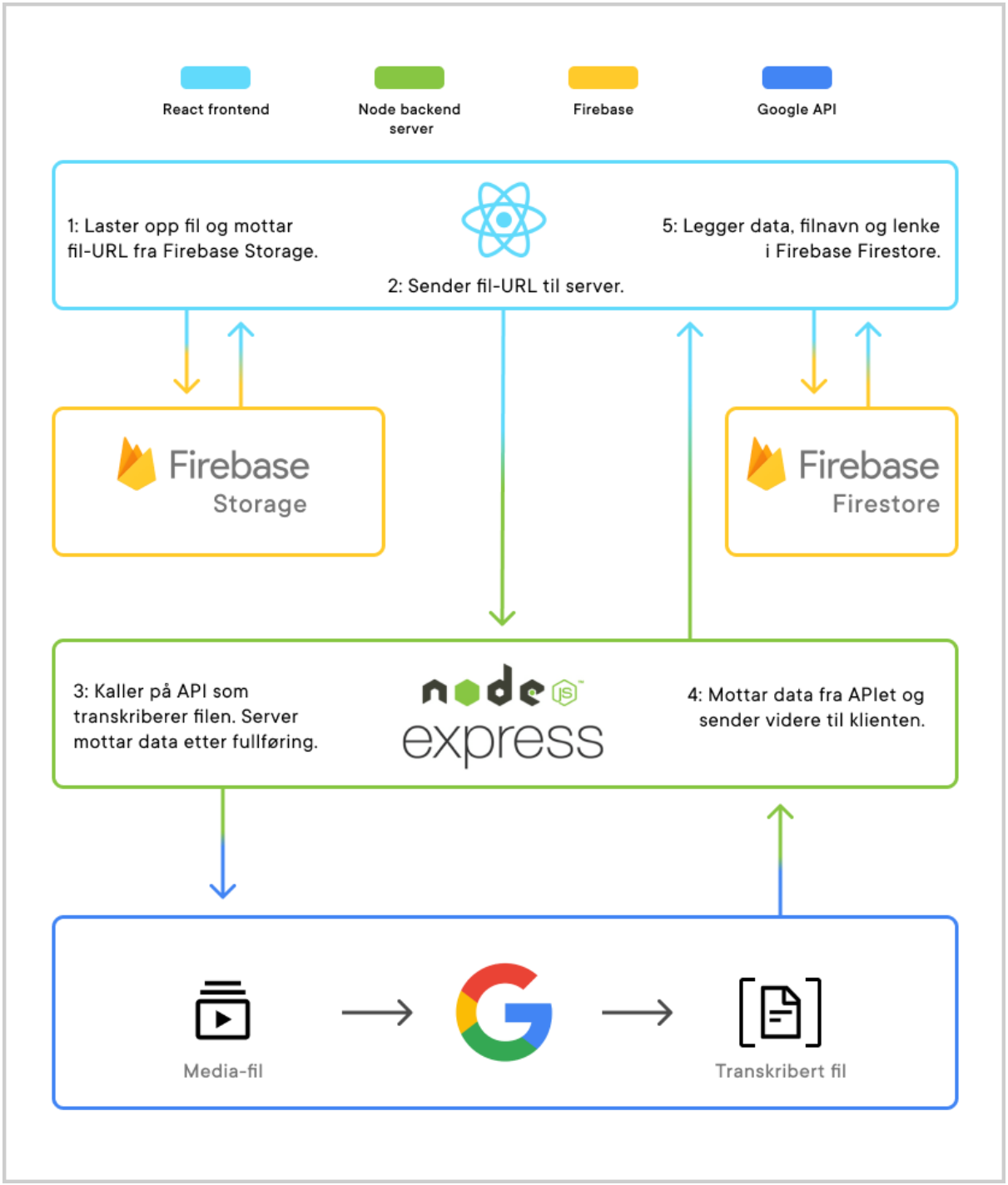


Figur 23: Skjermdump av e-post mottatt fra løsningen.

I figur 23 ser man at det blir inkludert navnet på filen du lastet opp, samt en lenke tilbake til siden, og det som ble oppsummert via tekstverktøyet.

4.1. Teknisk løsning

I denne delen av rapporten går vi gjennom den tekniske løsningen og hvordan de forskjellige teknologiene kommuniserer med hverandre. Den tekniske løsningen vår er bygget opp av en Node.js-server ved bruk av rammeverket Express for backend og JavaScript ved bruk av biblioteket ReactJS for frontend. Disse kommuniserer med et Google Cloud Speech-to-Text-API for transkribering og Google Firebase som database for lagring av filer og data om transkribering. “Application Programming Interface (API) er et programmeringsgrensesnitt som brukes for å utveksle data mellom to forskjellige applikasjoner” (Skjørten, 2019).



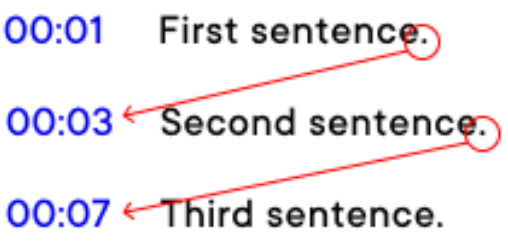
Figur 24: Flytskjema for løsningen.

4.1.1. Speech-To-Text

Vi brukte mye tid på å undersøke og forstå ulike API-er som kunne hjelpe oss med å transkribere lyd- og videomateriale. Til slutt landet vi på Google sitt Cloud Speech-to-Text (tale-til-tekst) API. Sammenlignet med de andre alternativene vi fant var dette veldig godt dokumentert og vi kunne bruke det gratis i 90 dager.

I stedet for å bare transkribere alt til én lang kontinuerlig setning uten noen form for punktsetting og grammatikk, kunne APIet også skjønne hvem som snakket (“Speaker 1:”, “Speaker 2:” osv.), når de snakket og hvor det skulle settes punktum på slutten av setninger. Med denne typen data hadde vi mye større frihet til å designe en mer brukervennlig og forståelig representasjon av det transkriberte materialet. Likevel har det sine begrensninger. Det er usikkert om det tar lenger tid å transkribere en fil siden vi har gratis bruker og tilgang.

Vi erfarte at gjenkjenning av hvem som snakket fungerte mindre optimalt. API-et kunne blande hvem som snakket flere ganger. For eksempel kunne “speaker 1” si en setning som egentlig var sagt av “speaker 2”. API-et kunne ikke fortelle oss når en ny setning begynte, bare en tid for hvert enkelt ord eller tegn. Tegnsettingen av punktum var mer tilfredsstillende enn gjenkjenningen av hvem som snakket, og traff som regel på slutten av hver setning. Derfor baserte vi logikken for tidsstemplene på punktumet i setningen før.



Figur 25: Tidsstempel basert på punktum.

Flere studier viser at akseptabel ventetid for å laste inn en nettside kan variere fra 1 til 41 sekunder. Det kan avhenge av erfaring, alder, individuell toleranse for å vente, type oppgave, forventet innhold, forventet lastetid og tilgjengelig informasjon om ventingen (Nah, 2003). Selv om lasting av nettsider ikke er det samme som å bruke et nettbasert verktøy, så kan det Nahs forskning brukes som en referanse for hvor tålmodige man er når man bruker Internett.

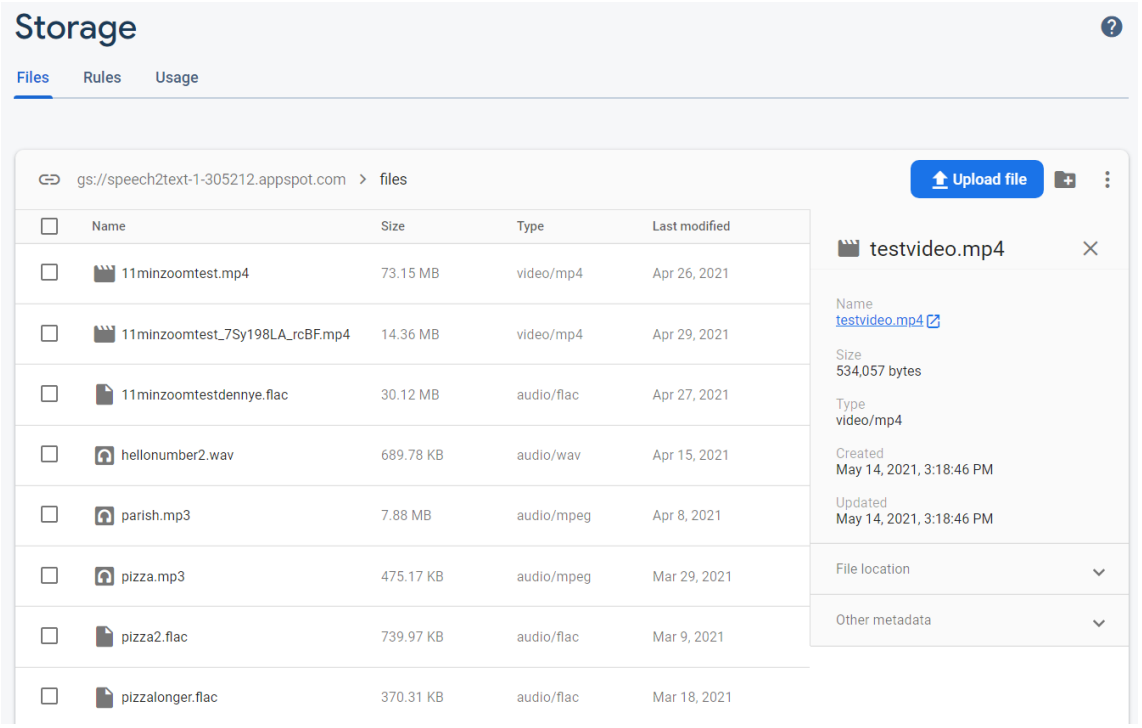
Et opptak på rundt 12 sekunder har som oftest tatt mellom 5-10 sekunder å transkribere. Et litt lenger på 30 sekunder har tatt rundt 15-20 sekunder. Opptak på 10 minutter har tatt omtrent fire minutter å transkribere, som trolig er greit for en MVP. Men i en reell situasjon, der en brukeren gjerne har et opptak på en time eller mer, kan det tenkes at man blir utålmodig av å vente på transkriberingen. Samtidig, som nevnt i kapittelet om funn fra brukertestingen, tar manuell transkribering ofte lang tid. Derfor er det rimelig å anta at en

automatisk løsning er et bedre alternativ for mange, da det tar kortere tid og man kan gjøre andre oppgaver samtidig.

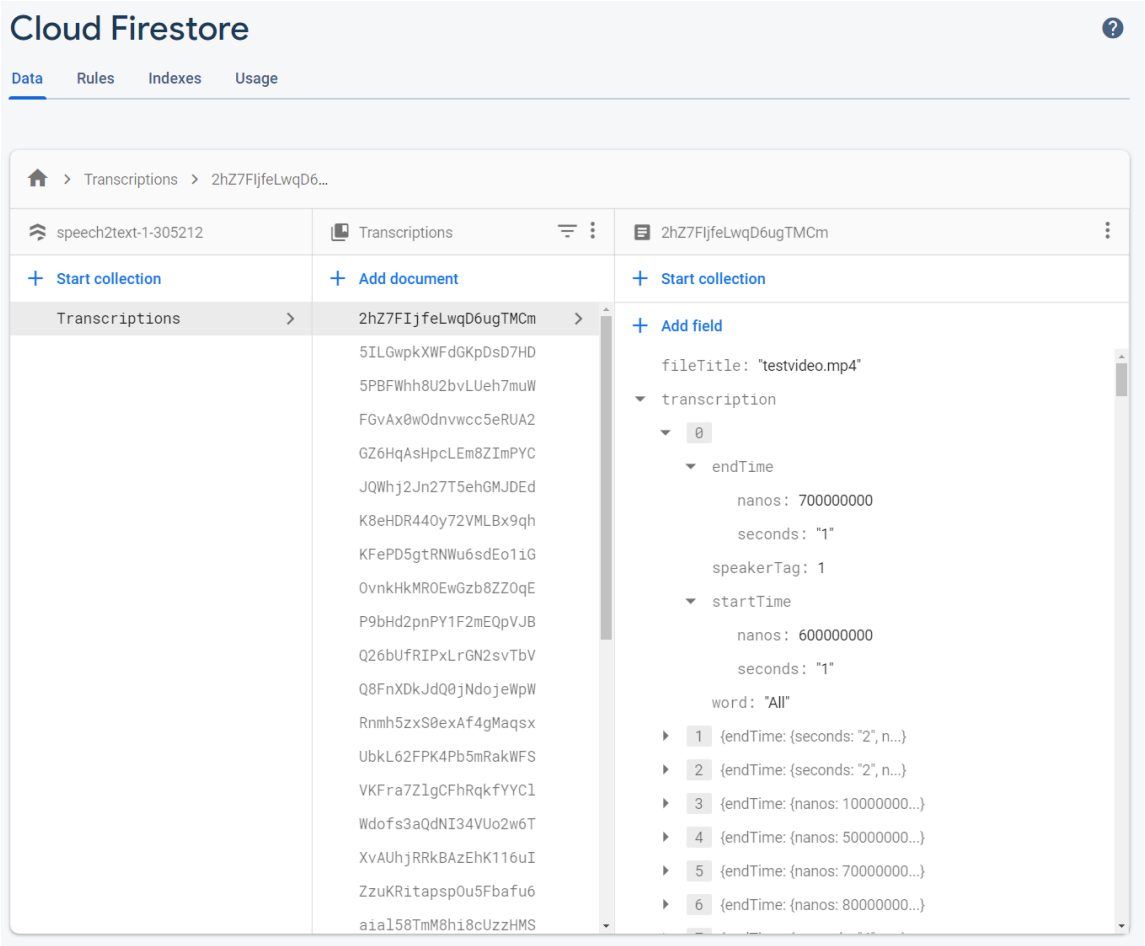
Ideelt sett hadde vi ønsket å få kontinuerlig tilbakemelding på statusen til transkriberingen, slik at bruker alltid visste hvor mye tid det gjensto. For noen vil det kanskje være for lenge å vente uten å ha en indikasjon på hvor lang tid det tar, men dette blir opp til den individuelle person. Et annet betalt API for konvertering av tale til tekst vil kanskje gi raskere resultater, men dette har ikke vi hatt tilgang til.

4.1.2. Firebase

For å i det hele tatt kunne servere en fil til API-et for transkribering, trengte vi en plass å lagre filene. Ettersom vi allerede var investert i Googles økosystem og Kodebyraaet hadde erfaring med bruk av Google Firebase, valgte vi det som skytjeneste for lagring av filer og database. I løsningen blir den valgte filen lastet opp til Firebase Storage slik at den kan transkriberes derfra. Videre bruker løsningen Firebase Firestore som database for lagring av data som for oss inkluderer den transkriberte lyden, filnavn og lenke til den opplastede filen.



Figur 26: Firebase Storage: Lagring av filer.



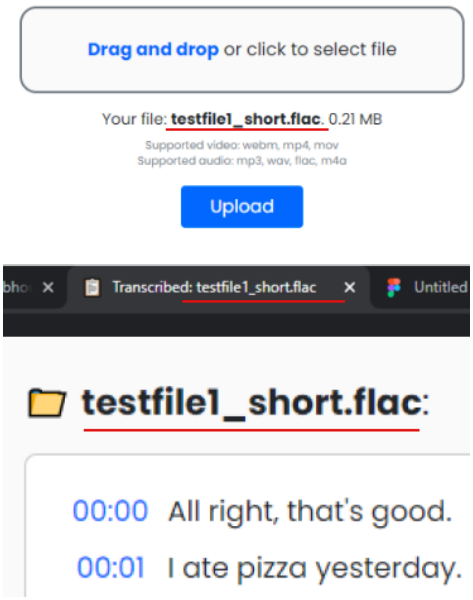
Figur 27: Firebase Firestore: Lagring av data.

4.1.3. React

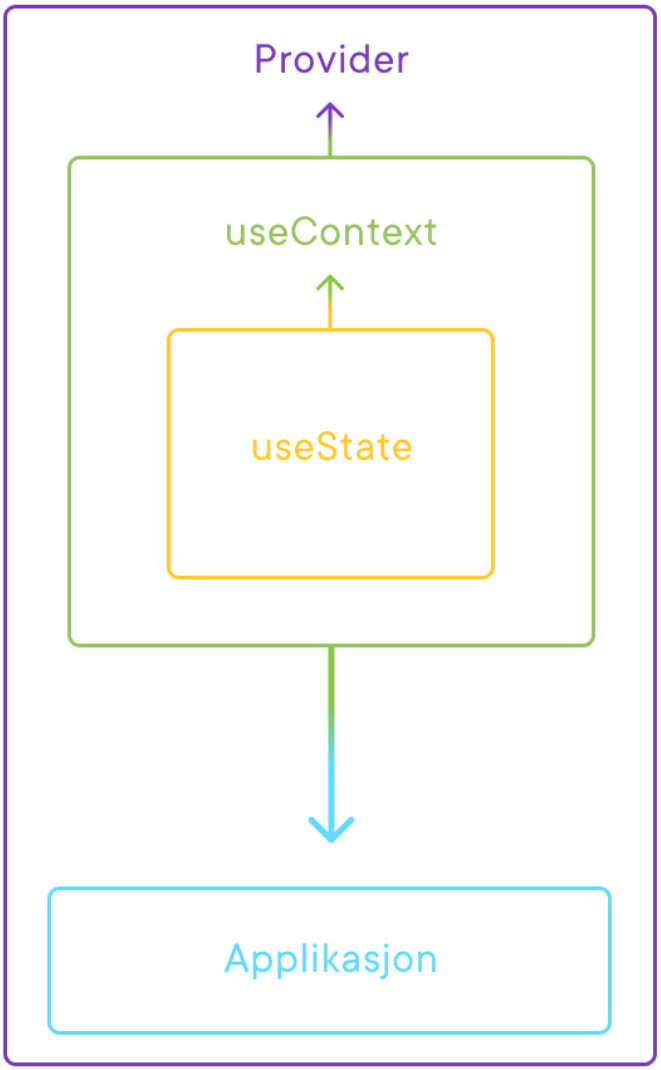
React er det er det JS-biblioteket vi har mest erfaring med å bruke og er et av de mest populære. Det er godt dokumentert og det finnes utallige ressurser tilgjengelig i tillegg til den offisielle dokumentasjonen.

React har siden 2019 gitt utviklere tilgang til “Hooks”. Vi har i hovedsak tatt i bruk useState for håndtering av tilstand, useEffect for å oppdatere data og useContext for å holde på tilstanden og spre data til hele løsningen. En Provider, eller forsørger, ligger rundt løsningen og gir alle komponentene tilgang til useContext-hooken. Ved å bruke disse funksjonskomponentene kunne vi håndtere tilstanden, oppdatere den og dele data i hele komponent-treet (alle elementene på alle sidene) (React, 2021). På denne måten kan vi vise data som er relevant for brukeren der vi ønsker. For eksempel ser man filnavnet til opptaket i nettleser-fanen og på siden med det ferdig transkriberte dokumentet uten å måtte hente

den samme informasjonen flere ganger. Dette fører til mindre repetisjon av kode og en raskere løsning (Se vedlegg I: Om React Hooks).



Figur 28: Skjermdump av løsning som viser samme informasjon på flere plasser med samme kode



Figur 29: Forklaring av dataflyten i frontend ved bruk av React Hooks.

4.1.4. Node.JS og Express

Vi har brukt Node.js og Express som serverløsning for vår backend. Node.js er en måte å kjøre JavaScript-kode utenfor nettleseren, men kan være omfattende og vanskelig å bruke. Express er et rammeverk som fungerer som et lag på toppen av Node.js-serveren som gjør det enklere å utvikle nettbaserte løsninger med Node.js. Node.js har gjort det mulig å kommunisere med Googles tale-til-tekst API (Hahn, 2016).

4.2. Design og utforming

Basert på brukertester kom vi frem til et utgangspunkt for designet og brukeropplevelsen. Under brukertestingen av prototypen fant vi mer ut av designets funksjonsgrad. Designet er enkelt og brukervennlig og har stort fokus på flyten av nettsiden. Da det er et brukerverktøy har vi utarbeidet korte beskrivelser av de første stegene slik at brukeren ikke skal bli forvirret. Et annet fokus har vært å gi brukeren gode tilbakemeldinger via skjermmeldinger etter å ha interagert med noe. Disse er tydelig merket med grønn, blå eller rød farge.

Vi har hatt stort fokus på at løsningen skal være lett å bruke. Dersom brukere ikke finner ut hva de skal gjøre eller har vanskeligheter med verktøyet, forlater de nettstedet (Nielsen 2012). Således har vi forsøkt å skape et verktøy som ivaretar kundenes krav og behov.

5. Diskusjon

I denne delen av rapporten presenterer vi våre egne vurderinger av prosjektet. Hvordan det har forløpt og hvordan vi ser på den tekniske løsningen som resultatet av forskning, prosess og metode.

5.1. Vurdering av prosess og metode

Vi har gjennom prosjektperioden brukt forskjellige verktøy for å forenkle prosessene og metodene. Litteratursøk har vært en sentral for å finne forskning relatert til prosjektet. I tillegg har vi i stor grad fått hjelp av brukertester til design og valg av laste-elementer. Bruk av Scrum og Kanban har vært sentralt for å gjennomføre prosjektet. Å dele inn i sprinter og å kunne visualisere arbeidsoppgaver har vært nyttig. Uten disse metodene og verktøy som Miro og Clubhouse hadde det i stor grad vært vanskeligere å komme i mål med prosjektet. Spesielt når vi har vært nødt til å ha hjemmekontor hele prosjektperioden.

5.1.1. Forskning

Ved hjelp av Google Scholar, IEEE og ResearchGate søkte vi etter og fant relevant forskning og litteratur. Vi har ved hjelp av diskusjoner i gruppen og sammen med intern- og ekstern

veileder funnet flere bruksområder hvor den kan vise seg å være nyttig. Det bidro til at vi kunne utvide horisonten for hva vi kunne forske på som var relevant for løsningen. Videre har det vært fordelaktig for å kunne bekrefte og avkrefte våre egne teorier rundt transkribering, menneskers tålmodighet og fokus.

5.1.2. Verktøy

Som det fremkommer av Miro-tavlen var de forskjellige ukene med GDS for mobil og web til stor hjelp. Vi trakk ut de idéene for design og funksjonalitet som vi antok var best. Disse kunne vi presentere for KB, hvor de kunne komme med videre innspill. Ved å bruke sprinter fra Scrum kunne vi sette ukentlige mål for forskning på teknologi og utvikling av løsningen. For hver sprint har vi fått reflektert på hva vi har gjort og hva vi har måttet fokusere på videre. Ved å hele tiden iterere og diskutere underveis har vi etter vår mening klart å begrense oss fra "scope creep". Scope creep er noe som skjer når et prosjekt vokser og man underveis får nye ideer og forslag til hva som kan være nyttig å ha med i prosjektet, og at de inkluderes i prosjektet (Jansen, 2018).

Vi har tatt vare på alt av ideer og funksjonalitet vi har kommet på underveis og skrevet de på Kanban-tavlen i Clubhouse. Både for å huske de til eventuell videre utvikling og i tilfelle vi hadde tid til overs.

5.1.3. Brukertester

Gruppen fikk godt utbytte av brukertesten vi hadde i sammenheng med design og brukeropplevelse. Vi hadde foretrukket flere deltagere, men med tanke på Covid-19 valgte vi å følge regjeringens råd for nærkontakt med andre. De vi testet hadde relevante jobber med hensyn til vårt prosjekt, da de regelmessig deltar i møter. For videre utvikling var det rimelig å anta at vi fikk tilstrekkelig med tilbakemeldinger.

Den andre runden med brukertesting tok for seg laste-animasjoner. Data vi samlet bidro til å ta valg for hvilke animasjoner vi skulle bruke for opplasting og transkribering. Basert på tilbakemeldingene fra brukertestene, skulle det helst vært benyttet bestemte animasjoner som viste progresjon fra 0-100 % begge steder i løsningen. Grunnet begrensninger fra

Googles API, kunne vi bare gjøre det på filopplasting. Med grunnlag i brukertestene anså vi spinne-animasjonen som den beste å bruke når man venter på en transkribering. Vi skulle gjerne fulgt opp brukertesten og utviklet en form for distraksjon/aktivitet som gjenspeilet det aspektet fra brukertesten, men med hensyn til tid ble vi enig om å prioritere mer fundamental funksjonalitet.

5.2. Vurdering av løsning

Når vi ser på løsningens endelige versjon, mener vi at design-sprintene med prototyping og brukertesting helt i starten var noe av det viktigste vi gjorde. Ved at vi gikk gjennom så mange iterasjoner i starten, var det sjelden tvil om hvordan den skulle se ut eller fungere når vi kom lenger ut i prosjektperioden.

5.2.1. Design

Grunnlaget for det endelige designet kom av flere lange prosesser, inkludert GDS, brukertesting og flere runder med prototyping. Det førte frem et design og en flyt i løsningen som vi mener vi kan redegjøre for, og være fornøyd med. Arbeidet med designet foregikk fra første dag, og er på et vis ikke ferdigstilt siden løsningen kan videreutvikles på sikt.

5.2.2. Teknisk

Vi hadde en svært teknisk oppgave og mesteparten av prosjektperioden gikk til å løse denne på best mulig vis opp imot kravene satt fra KB. Vi har tatt til etterretning at de tekniske verktøyene vi har brukt er relevante for arbeidsmarkedet og for fremtiden. Som for eksempel Cypress til testing og Firebase for skylagring. Parprogrammering var et viktig verktøy for bacheloroppgaven og sørget for at vi kunne levere en god løsning og god kode.

Løsningen har stort potensiale for videreutvikling og kvalitetssikring. Basert på vår eksisterende kunnskap og den vi har tilegnet oss underveis, har vi valgt teknologier og funksjonalitet med omhu for ikke å sitte igjen med en halvhjertet løsning.

5.3. Vurdering av vår forankring i forskning

Selv om den opprinnelige tanken bak prosjektet var ment for noen som ofte er i møter med begrenset tid imellom, har vi tilsynelatende klart å lage et verktøy som også kan fungere i andre situasjoner. Det startet med at vi undersøkte hvordan menneskers fokus kan variere i møter og verktøy som kan være behjelpelige med det. Samt verdien av å ha notater fra møter i etterkant. Det ble etter hvert naturlig å ta høyde for hvordan løsningen kunne fungere for enkelte med nedsatte funksjonsevner som for eksempel dårlig hørsel uten at det ville kreve mye ekstra utvikling. For mange har hverdagen blitt ganske annerledes enn den var for 15-20 år siden. Dermed er det grunn til å anta at deler av forskningen vi fant potensielt er utdatert. Det vil nok etter hvert dukke opp nye studier som kan være enda mer nøyaktige og med flere deltakere siden det nå er så mange som arbeider digitalt og via nett.

5.4. Nytte for oppdragsgiver

Oppdragsgiver har uttrykt glede over at vår løsning er noe de kunne ha brukt til vanlig etter sine møter. Det blir opp til KB om de vil bygge videre på løsningen, med eller uten vår hjelp. Basert på deres tilbakemeldinger kan det se ut som de ser potensiale i løsningen vi har presentert.

5.5. Vurdering av gruppens mål opp mot resultat

Målet vårt med bacheloroppgaven om å tilfredsstille KBs visjon og la alle i gruppen få bruke sin kreativitet og kunnskap, har vært viktig for oss. Enkelte har fått utfoldet seg på design, mens andre har fått mer utløp når det kommer til det tekniske. Vi har fått brukt mye av det vi har lært i løpet av studiet. Det har vært interessant å arbeide med denne utfordringen over en lengre periode.

5.6. Videre utvikling

KB har trolig selv mange forslag selv til hvordan løsningen kan bli bedre. Antageligvis har vi gjort rede for den mest relevante funksjonaliteten vi hadde utviklet om vi hadde mer tid og ressurser. Vi ser stort potensiale for en slik løsning i dagens situasjon.

Vi fikk underveis i prosjektet mange ideer til mulige utvidelser av løsningen. Siden fokuset lå på å levere en MVP, måtte nyttige, men ikke nødvendige funksjoner vente. Både gruppen og KB hadde forslag til funksjonalitet som ble utelatt fra løsningen. Vi hadde møter hvor vi avklarte prioriteten til disse og arbeidet ut fra dette. Videre ble de sortert ut som "nice to haves", eller *fint å ha som ikke er nødvendig*. Det var viktig å begrense iterasjonene underveis og heller notere seg hva som kunne bli mulige utvidelser i fremtiden. I denne delen av rapporten utdyper vi hvordan vi ville videreutviklet løsningen.

5.6.1. Integrering med andre møteverktøy

Ved å kunne ha løsningen integrert med de største aktørene innen digitale møter som Zoom, Microsoft Teams eller Google Meet, ville løsningen transkribert møtet etter det var ferdig og deretter sendt transkriberingen via e-post. Videre kunne man klikket seg inn på lenken i e-posten for å lage sammendrag og høre gjennom opptaket.

5.6.2. Brukere og administrator

Et godt tilskudd for brukerne av applikasjonen ville antakelig være å kunne logge inn og ha en oversikt over sine møteopptak og transkriberinger. Der kunne alt vært samlet på én plass og være tilgjengelig for de involverte. Vi har sett på løsninger der selskaper velger en administrator som så kan holde orden på tilgang til funksjoner for de ansatte. Dette åpner for en rekke muligheter når det kommer til samarbeid om redigering, signering av møtedeltakere, deling m.m.

5.6.3. Bedrift-system

Et team-basert system for bedrifter kan samle alle møter på ett sted. Her kan for eksempel én på et team dele møtene med transkribering og referat med de som er til stede. Resten av teamet har da mulighet for å gå inn og lytte, redigere og lese referat av tidligere møter. Et annet scenario kan være at noen fra andre team i bedriften har deltatt på et møte. Man kan da legge til spesifikke deltakere fra møtet som kan finne igjen møtet i en kalender. Møtet kan da åpnes på samme måte som i det forrige eksempelet. Et slikt system kan potensielt effektivisere hvordan møter blir håndtert av en bedrift.

5.6.4. Automatisk oppsummering

Noe som kunne vært nyttig for å spare enda mer tid, er en funksjon som lager oppsummering av transkripsjonen for brukeren. Vi har sett eksempel på slike verktøy i vår research, men mange er i startfasen av utvikling. De fleste bruker kunstig intelligens for å lage sammendrag av teksten. Faren med dette er at om deler av transkriberingen ikke er hundre prosent korrekt kan teksten mistolkes og oppsummeringen vil potensielt mangle viktig informasjon. På lik linje er det usikkert at selv om transkriberingen er helt riktig så vil sammendraget bli riktig.

5.6.5. Ikke bare bedriftsmøter

Vi har innsett at bruken av løsningen vår kan være nyttig for flere enn bare de som sitter i bedriftsrelaterte møter. I litteratursøk, hvor vi har sett etter bruken av løsningen for mennesker med nedsatt funksjonsevne, har vi også sett at studenter i høyere utdanning og forfattere som driver med kvalitativ forskning, ofte gjør transkriberinger av sine intervjuer når de samler data (Bakke 2016).

5.6.6. Sikkerhet

Løsningen er ikke sikret med hensyn til personvern. Ingenting er kryptert og alle transkriberinger er åpne for alle som har lenken. Dette er noe vi ville arbeidet med hvis den skulle bli et faktisk produkt. Krypterte lenker som ikke kan indekseres (finnes av søkemotorer som Google), passordbeskyttelse, invitasjon basert på e-post og innlogging er faktorer vi ville vurdert for sikkerhet og kvalitetssikring av løsningen. Vi har for løsningen på dette stadiet ikke sett på det som en viktig del å implementere, men vi vil understreke at det ville vært øverst på listen for videreutvikling.

5.6.7. Økonomisk

Med hensyn til hvor mye tid og ressurser som potensielt kan spares på å bruke et slikt verktøy, vil det være i KBs interesse at det kan generere inntekter. Hvordan det kan løses blir deres utfordring, men Software as a Service (Saas) kan være en mulighet.

> Programvare som tjeneste (software as a service - SaaS), som er en modell for leveranse over et nettverk hvor kunden benytter leverandørens applikasjon(er) på en nettsky-infrastruktur. Kunden har i utgangspunktet ikke kontroll over verken applikasjoner, nettverk, servere, operativsystemer eller lagringsmuligheter (Skytjenester, 2021).

Tjenester som Microsoft Office 365, Google-applikasjoner, Dropbox og Slack er eksempler på SaaS (Vladimirskiy, 2016) og dette antar vi kan være en god økonomisk modell. Vi håper i tilfelle at de finner en som vil gjøre løsningen attraktiv for så mange som mulig. Enten det gjelder enkeltpersoner, bedrifter eller institusjoner.

6. Konklusjon

Hvordan kan man gjøre opptak av digitale møter mer verdifulle?

Vi har i løpet av prosjektet utviklet en løsning som lar brukere enkelt transkribere lyd- eller videofiler, redigere teksten og dele den via e-post. Gjennom sprinter og brukertesting har vi kommet frem til et design og en funksjonalitet som svarer til Kodebyraaets ønske for en MVP. Et verktøy som kan gjøre en hektisk hverdag med møter enklere.

Det er usikkert om løsningen kan fullstendig erstatte notering i møter. Vi antar imidlertid at den kan brukes som et verktøy for å støtte notering. Samtidig vil man ved å lagre opptakene og teksten fra transkriberingen i skyen, enklere kunne gå tilbake til opptakene av møtene. Det er spesielt på dette området vi gjerne skulle hatt tid til å utvide til en slags team-basert løsning hvor alle deltagere fra et møte enkelt kunne hatt tilgang til sine tidligere møter og transkriberinger.

Covid-19-pandemien har økt forekomsten av digitale møter og det er derfor grunn til å anta at problemstillingen vår bare har blitt mer relevant enn før. Automatisk transkribering av digitale møter kan bidra til at man ikke trenger å ta notater underveis. Man kan holde fokus på møtet og hva som blir sagt, og i tillegg være åpen for å kunne delta i samtalen selv. Har

man en travel hverdag med mange møter, kan man gå rett fra ett møte til et annet, og selv velge når man vil bearbeide opptakene. Med mulighetene for å se eller lytte på opptaket og redigere transkripsjonen, kan man kvalitetssikre at referatene blir korrekte før de kan deles med de det skulle gjelde. På bakgrunn av dette føler vi at vi svarer på problemstillingen ved at opptakene blir mer verdifulle når man kan transkribere, redigere og dele; alt i ett verktøy.

Bachelorprosjektet har vært utfordrende og lærerikt. Vi har brukt mye av kompetansen vi har tilegnet oss i løpet av tre år på Høyskolen Kristiania. Samtidig har det vært spennende å jobbe for en reell arbeidsgiver og vi setter pris på den innsikten det har gitt oss.

7. Litteraturliste

Alexandros, B. (2017) The Design Sprint (And Three Reasons Why It's a Solid Win). Medium. Hentet fra: https://balexandros.medium.com/the-design-sprint-53a9d2648e92 (Lest 24. mars 2021).

Bailey, J. (2008) First steps in qualitative data analysis: transcribing. Family Practice, 25(2), s.127-131. Hentet fra: https://academic.oup.com/fampra/article/25/2/127/497632 Lest 24. mai 2021.

Bakke, Helene L. (2016) Videregående elever med hørselsnedsettelse. En kvalitativ studie av *deres behov i skolen*. Universitetet i Oslo. Hentet fra: http://hdl.handle.net/10852/52084 (Lest 21. mai 2021)

Cadle, J. og Yeates, D. (2008) *Project management for information systems*. 5th ed. Harlow [etc.]: Pearson/Prentice Hall.

Datatilsynet. (2021) Skytjenester. Hentet fra: https://www.datatilsynet.no/personvern-pa-ulike-omrader/internett-og-apper/skytjenester/ (Lest 26 mai 2021).

En.wikipedia.org. 2021. Minimum viable product - Wikipedia. Hentet fra: https://en.wikipedia.org/wiki/Minimum_viable_product Lest 27. mai 2021.

Granulo, A. og Tanovic, A. (2019) *Comparison of SCRUM and KANBAN in the Learning Management System implementation process*. 2019 27th Telecommunications Forum (TELFOR). Hentet fra: https://ieeexplore.ieee.org/document/8971201 (Lest 31. mars 2021).

Guides.github.com. (2021) Hello World · GitHub Guides. Hentet fra: https://guides.github.com/activities/hello-world/ (Lest 13. mai 2021).

Hahn, E., (2016). Express in action. Shelter Island, NY: Manning Publications.

Interaction Design Foundation. (2020) Using Color to Confuse and to Prevent Unwanted Actions. Hentet fra: https://www.interaction-design.org/literature/article/using-color-to-confuse-and-to-prevent-unwanted-actions#:~:text=Color%20can%20be%20used%20to,on%20your%20favorite%20design%20guru (Lest 29. mars 2021).

Jansen, A., 2021. IKT-servicefag Vg2 - *Hva er spesielt for IKT-prosjekter?* - NDLA. ndla.no. Hentet fra: https://ndla.no/nb/subject:25/topic:1:193104/topic:1:119652/resource:1:119659?filters=urn:filter:9d6d3241-014d-4a5f-b0bc-ae0f83d1cd71 Lest 24. mai 2021.

Kodebyraaet. (2021) Hentet fra: https://kodebyraaet.no/ Lest 11. mars 2021.

Magnussen, Jo Christian. (2010) Prototypes in usability testing: the implications of richness in interaction fidelity. Universitetet i Oslo. Hentet fra: https://www.duo.uio.no/bitstream/handle/10852/8746/Magnussen.pdf?sequence=4&isAllowed=y (Lest 15. mai 2021)

Matheson, J., (2015) The Voice Transcription Technique: Use of Voice Recognition Software to Transcribe Digital Interview Data in Qualitative Research. The Qualitative Report. Hentet fra: https://nsuworks.nova.edu/tqr/vol12/iss4/1/ Lest 13. mai 2021.

Miaskiewicz, T. og Kozar, K., (2011) Personas and user-centered design: How can personas benefit product design processes?. Design Studies, 32(5), s.417-430. Hentet fra: https://doi.org/10.1016/j.destud.2011.03.003 Lest 20. mai 2021.

Mollick, E. og Rothbard, N., (2013) Mandatory Fun: *Gamification and the Impact of Games at Work*. SSRN Electronic Journal,. Hentet fra: https://poseidon01.ssrn.com/delivery.php?ID=753074071124064070014114107122095006010045004048003005075122089087089078095003085029126119017036023013055086071064022071067089045045047076049099088100069095019010006081012008124002068

670710690290990041250930080900290230820151120020921060030660030841 19&EXT=pdf&INDEX=TRUE (Lest 24. mai 2021)

Muşlu, K., Brun, Y., Holmes, R., Ernst, M. og Notkin, D., (2012). Speculative analysis of integrated development environment recommendations. ACM SIGPLAN Notices, 47(10), s.669-682. Hentet fra: https://dl.acm.org/doi/pdf/10.1145/2398857.2384665?casa_token=SZ3Udb2McAYAAAAA:gVtorLwoWNKQDGY2qwbx2tuhBPbAhf0nanI9-vvKZinWrDeEwKgKVoWXnBXuzL6GH2s1MHaTGvLs Lest 3. mai 2021.

Nah, F. (2003) A Study on Tolerable Waiting Time: *How Long Are Web Users Willing to Wait?* Hentet fra: https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1751&context=amcis2003 (Lest 14. mai 2021)

Nielsen Norman Group. (2012) Usability 101*: Introduction to Usability*. Hentet fra: https://www.nngroup.com/articles/usability-101-introduction-to-usability (Lest 31. mars 2021).

Otter Voice Meeting Notes. (2021) Otter Voice Meeting Notes. Hentet fra: https://otter.ai/ (Lest 2 juni 2021)

Pereiro, F. (2021) Declarative Programming: *Is It A Real Thing?*. Toptal Engineering Blog. Hentet fra: https://www.toptal.com/software/declarative-programming (Lest 13. mai 2021).

Rawsthorne, D., & Shimp, D. (2018) Scrum Handbook: Single-Team Scrum. CreateSpace Independent Publishing Platform.

Reactjs.org. (2021) React – A JavaScript library for building user interfaces. Hentet fra: https://reactjs.org/ (Lest 13. mai 2021).

Reactjs.org. (2021) Hooks at a Glance – React. Hentet fra: https://reactjs.org/docs/hooks-overview.html#:~:text=Hooks%20are%20functions%20that%

20let,you%20use%20React%20without%20classes.&text=You%20can%20also%20create%20your,stateful%20behavior%20between%20different%20components (Lest 13. mai 2021).

Richter, H., Abowd, G., Geyer, W., Fuchs, L., Daijavad, S. og Poltrock, S. (2001) Integrating Meeting Capture within a Collaborative Team Environment. Ubicomp 2001: Ubiquitous Computing. Hentet fra: https://www.researchgate.net/publication/27521353_Integrating_Meeting_Capture_within_a_Collaborative_Team_Environment. (Lest 20. mai 2021).

Rubin, K. (2013) *Essential Scrum*. Upper Saddle River, NJ: Addison-Wesley, s.7.

Ruparelia, N., (2010) The history of version control. *ACM SIGSOFT Software Engineering Notes*, 35(1), s.5-9. Hentet fra: https://dl.acm.org/doi/pdf/10.1145/1668862.1668876?casa_token=bWM5yjkRHgUAAAAA%3AP_PgVMQzfcDvIM4Y6VJt2bxc0YPFNBqJDAN6VHhV7NfyM6GyhbPtsOPzwk5YbU1BzGKdB8RjdrH2 (Lest 31. march 2021).

Selbekk, K. og Gjøby, S., (2021) Slik bruker du nye React Hooks. Kode24.no. Hentet fra: https://www.kode24.no/kodenytt/slik-bruker-du-nye-react-hooks/70408478 (Lest 16. mai 2021).

Skjørten, T., (2019). Hva er API? og 9 andre spørsmål og svar om API. Visma Blogg - om teknologi, regnskap, skatt, lønn, innkjøp, HR. Hentet fra: https://www.visma.no/blogg/hva-er-api-sporsmal-og-svar/ Lest 3. mai 2021.

Sutherland, J., & Schwaber, K. (2007) The scrum papers. *Nuts, Bolts and Origins of an Agile Process*.

Vijayasarathy, L. og Butler, C. (2016) Choice of Software Development Methodologies*: Do Organizational, Project, and Team Characteristics Matter?*. IEEE Software, 33(5), s. 86-94. Hentet fra: https://ieeexplore.ieee.org/document/7006383 (Lest 30. mars 2021).

Vladimirskiy, V., 2016. 10 Popular Software as a Service (SaaS) Examples - Nerdio Academy. Nerdio. Hentet fra: https://getnerdio.com/academy/10-popular-software-service-examples/ Lest 1. juni 2021.

Web Content Accessibility Guidelines 2.0, W3C World Wide Web Consortium Recommendation (2021) Hentet fra: http://www.w3.org/TR/WCAG20/ (Lest 12. mai 2021).

Williams, L., Kessler, R., Cunningham, W. og Jeffries, R. (2000) Strengthening the case for pair programming. IEEE Software 17(4), s.19-25. Hentet fra: https://collaboration.csc.ncsu.edu/laurie/Papers/ieeeSoftware.PDF (Lest 20. mai 2021).

Whittaker, S., Laban, R. og Tucker, S. (2006) Analysing Meeting Records: An Ethnographic Study and Technological Implications. *Machine Learning for Multimodal Interaction*. Hentet fra: https://www.researchgate.net/publication/221040328_Analysing_Meeting_Records_An_Ethnographic_Study_and_Technological_Implications. (Lest 20. mai 2021).

Østerbæk, A. (2019) Why design sprints are overrated. Medium.Hentet fra: https://medium.designit.com/why-design-sprints-are-overrated-3f0afdb0d0ea (Lest 24. mars 2021).

7.1 Verktøy:

Clubhouse: https://clubhouse.io/

Cypress: https://www.cypress.io/

Discord: https://discord.com/

ExpressJS: https://expressjs.com/

Figma: https://figma.com/

Git: https://git-scm.com/.

Github: https://github.com/

Miro: https://miro.com/

Node.js: https://nodejs.org/en/

Vedlegg

Vedlegg A: Arbeidskontrakt

Denne arbeidskontrakten bygger på et sett med typiske mål, oppgavefordelinger, prosedyrer og retningslinjer for interaksjoner for studentarbeid. Vi benytter denne for å ha en felles enlighet og klarhet i hvordan samarbeidet skal utarte seg og dermed legge opp til god samarbeid og trygget for alle i gruppen.
Vi ønsker en løsning som både gruppen og oppdragsgiver er fornøyd med. I tillegg vil vi levere noe vi som gruppe kan være stolte over.

Medlemmer

Bernt Johan Aspehaug, Martin Molvær, Per-Kristian Vinje

Mål

Effektmål

Vi ønsker å trene på effektivt gruppearbeid, som i alle høyeste grad vil være nyttig både for videre studier og i arbeidslivet. For å oppnå dette vil vi i fellesskap legge en plan med konkrete mål og oppgaver som skal løses, samt bruke en rekke verktøy for utvikling og kommunikasjonsflyt. Underveis i arbeidet vil vi gjøre regelmessige vurderinger av hvordan dette fungerer for oss, og hvilke eventuelle justeringer vi kan gjøre for å samarbeidet så effektivt som mulig.

Resultatmål

Gruppen er enige om å oppnå en bestemt karakter: B eller bedre. Gruppen ønsker også å sitte igjen med noe som bedriften kan bygge videre på til et omfattende, reelt produkt.

Roller

Alle på gruppen er med på planlegging, avgjørelser og utvikling.

Prosedyrer

Kommunikasjon i gruppen

Alle medlemmer av gruppen er i en felles kanal på Discord. Det er plattformen vi i gruppen er best kjent med og er derfor også den mest pålitelige for oss. Her kan vi ha videosamtaler, chatter for diverse temaer og dele filer med hverandre som også er enkelt å finne igjen senere. Gruppen er også på Slack hvor vi kan kommunisere med bedriften.

Avgjørelser

Alle avgjørelser skal tas i fellesskap. Ved uenighet avgjør flertallet.

Møteinnkalling/felles arbeidstid

Alle skal vise seg på kamera og legge en plan for dagen kl 0900 fire dager i uken. Hvilke dager gruppen møtes kan variere , og neste arbeidsdag skal være avtalt før man avslutter for dagen.

Fravær/forsinkelser

Ved forsinkelser eller fravær skal dette varsles i god tid i gruppechat.

Arbeidsmengde

Alle i gruppen skal legge ned arbeidsmengden som skal til for å nå våre mål og at alle viser engasjement underveis. Om enkelte har mindre arbeid i perioder er det viktig å tilby seg å hjelpe. Dersom noen i gruppen har problemer med å utføre sine oppgaver skal dette meldes fra om slik at man kan omfordele oppgaver. Det viktigste for oss i gruppen er at alle møter opp når de har mulighet og yter sitt beste.

Brudd på arbeidskontrakten

Ved brudd på regler kan det i verste fall føre til eksklusjon fra gruppen. Dette er noe som skal begrunne nøye. Ved brudd på punkter i kontrakten skal følgende prosedyre følges: Skriftlig advarsel gis til vedkommende med en frist vedkommende har til å forbedre seg. Ved

fristens utløp tas saken opp i plenum, der gruppen kan avgjøre hvilke videre steg som skal tas. Endringer som kan bli gjort er for eksempel omrokkering på oppgaver. Eventuelle konsekvenser blir avgjort av gruppen i fellesskap.

Regler

1. Ingen spørsmål er dumme
2. Si fra om du kommer for sent.
3. Alle skal lytte til andres innspill og ideer.
4. Vær konstruktiv med kritikk og tilbakemeldinger.
5. Om en beslutning er vedtatt av flertallet skal denne beslutningen respekteres.
6. Ta hensyn til andre utfordringer. vær snill og hjelp hverandre.
7. Innsats teller mest

Underskrifter

Sted:_____Oslo_____ Dato:____15.01.21_____

……...

Tilbake til teksten her.

Vedlegg B: Møtereferater

Under ligger det vi anser som de viktigste møtereferatene som har vært med på å forme løsningen:

Første møte med Kodebyraaet 12.01.21

Mobilapplikasjon

- Bakgrunn: Tanken bak: “Ta opp et møte og sette kapittel. Slippe å miste fokus på møtet. ”

- Mulighet for å transcribe?
- Software as a service?
- Applikasjon med tilhørende nettside?
- Sette markør, navngi senere.
- Dele via dropbox/cloud-tjenester
- GDS?
- MVP?
- Tenke API tidig for plattformuavhengighet
- Finne suksessfaktorer og ta med videre til tjenestebeskrivelse
- Research: Kan man tappe. Strømbruk. Tilgjengelige funksjoner. Viktige ting rundt bruk og gjennomføring.
- Miro?

- Krav:
- MVP?
- Løsning?

Forventninger:
- TO-sidig

1: tjenesten/appen

2: lærer av hverandre oss/kb

3: tilfører vår kunnskap og evne til å løse det som skal løses. Finne gode løsninger sammen.

4: gode prosesser

5: vi leverer noe som kan brukes

6: håper å ha en app som kan brukes

Emils tanker:

- Enkel app på mobil
- Ta opp et møte og kunne sette inn kapittel der du vil
- Tap tre ganger, så setter du inn et kapittel for 30 sek siden
- Stemmegjenkjenning

- Enterprise for større selskaper
- Frittstående tjeneste, som kan kobles på diverse plattformer
- Stemmegjenkjenning, som gir et manus “martin sa: “”, PK svarer: “”, etc”
- Software as a service - koble på en skytjeneste
- Bruke design-drevet utvikling
- API kan være lurt for videre utvikling.
- React Native på alle apper de lager
- Finne viktigste suksessfaktorer og ta de inn i tjenestebeskrivelsen.

Krav til funksjonalitet:

- Opptak
- Kapittel
- Tapping for innsett av kapittel
- Inkognito skjermbilde

Forventninger

- Forventinger til løsningen
- Forventing om at vi kan lærer av de og at de kan lærer av oss
- Skal ikke være en kunde som kommer med en kravspesifikasjon og ikke er involvert
- Håper det blir en app Emil kan bruke når vi er ferdig.

Møte med Kodebyraaet 20.01.21

Til stede:

Fra gruppe 36: Martin Molvær, Per-Kristian Vinje, Bernt Johan Aspehaug.

Fra Kodebyraaet: Johan Josøk, Emil Bonsaksen.

- Sprinter innad i utviklingsprosessen?
- Stykke opp utviklingen. Ha noe kodet tidligere, iterere på koden.
- Få noe ut tidlig å teste.
- Først da får man kvalitativ feedback.
- Finn noe som konkretiserer hva vi bygger.
- Brukerhistorier er viktig. GRANULÆRT. Håndfast.

Legge en god plan for produktutvikling. Prioritere funksjoner etc. Verktøy: Clubhouse.

- Kanskje få til en prototype tidligere enn uke 7
- Planlegge for programmering tidligere. Mindset.
- Prototype for produktet også.
- MVP, men også en liste over ALT vi kommer på som funksjoner til appen. (Viktig læring) Kan også vise om vi klarte å få til flere funksjoner enn det vi hadde tenkt.

Output:

4: Utfordrer oss til å lage prototype denne uken

Basic, ikke nødvendigvis klikkbar - ikke tenk design.

5: Design og utvikling av klikkbar prototype. (Jobbe med flere versjoner, iterere på versjoner ikke erstatt) Iterasjon.

6: Kvalitative undersøkelser og feedback.

7: Iterasjon. Skrive brukerhistorier, få ned features.

8 - 10:

10-15:

User stories: Gi tilbakemelding på.

Bli enig med et sett med features som vi blir enig om er MVP-en.

Droppe at vi skal lage en app. Tenke mer features.

Recorde ZOOM eller whereby, SaaS-bit.

Hva kan vi tjene penger på?

Transkribere møter, systematisering av recordings og deling av recordings.

Lage en tjeneste som kobler seg på disse møte-appene, som systematisere og deler.

Web-motpart som man kunne legge kapitler på.

Dele med alle i møtet.
Lage noe for en verden med flere digitale møter enn noen gang...
Et API som kan ta i mot fra hvor som helst, som en del av løsningen så e det et applikasjonsgrensesnitt, som et startpunkt for organisering på web.
Mange muligheter å hekte på etterhvert, etter man har etablert MVP.

Voice til tekst...?
Mulighet til å komprimere teksten til det viktigste.

Webapplikasjon. Anvendelig.

Vi har opptaket, hvordan kan vi bruke den til å sette kapittel, distribuere.
Kanskje ikke tenk på opptak. En database med lydopptak. Utgangspunktet webm, legge inn lydopptaket. Hva kan vi gi softwaren som gir verdi til opptaket? Sette kapitler, kanskje speech to text.

Se på integrasjoner mot de største ZOOM WHEREBY MEET TEAMS ETC.
Nice to have: automatikk. få filene inn i verktøyet automatisk. Fokusere på å ha opptaket lokalt totalt.

Potensialet er større, jobbe meir med tjenesten enn selve medie.(Organisering/Systematisering/Sette kapitler).

Endre ukene. Kjøre 2 GDS og prototyper??
Wherbyrecordings, lage kapitler
Korte ned prototyping.

Fokusere på brukerproblem ikke spesifikk løsning!!! Viktigste med prosessen er iterasjoner ikke utfall. Komme frem til løsninger som man har trua på!

Møte med intern og ekstern veileder 27.01.21

- Realitet: Folk kommer ikke til å høre på lyd
- Sammendrag av høydepunkter blir viktig
- Kommer an på oss og hva vi vil lære

For lansering: Minst mulig features. Mest mulig komplett. Bruke firebase for autentisering og brukerhåndtering.

!Trenger ikke håndtere teams i første omgang.

!Gratis med enkeltbruker.

!Laste opp opptak i første omgang.

Mulig prioriteringsliste:
1 Fra lyd til tekst som kan deles.

2 Ha en egen bruker og se tidligere med url.

3 Teams.

4 Kunne ta opp lyd via appen.

Pers tanker:
Å tappe inn på lyd er farlig med hensyn til personvern når ikke alle kan vite om det.
Kommentar: Kompetanse på team som skal kunne levere stor tjeneste som gjør “alt” mtp native features og OS-spesifikke egenskaper?

Jobba på og sette sammen noe konkret som skal virke
like verdifullt som

Sette sammen noe som “blir som det blir”, men man tar med seg flere erfaringer og kunnskap.

Faglig forankring:
Knytte det vi gjør og begrunne det til noe mer enn “dette var lurt”
Forankre det i noe og knytte til fagfelt.
Både det vi lager og prosessen vi er gjennom for å lage det.

Problemstilling:
Knytte problemstilling opp mot prosjektet.
Nåværende er fin så langt. Kan finpusses senere.

React+firebase = bra.
Vi får jobb hvis vi kan disse!

Clubhouse -> iterations.
Ikke henge seg opp i timer.
Bevisst på hvor stor oppgaven er.

Definere perioder og putter ønskelige features i den.
2 uker utvikling
En uke med testing osv
Finne ut hvor mange story points vi trenger.
Hvor mange klarte vi, hvor mange trenger vi

Epics: Ikke tids-spesifikk. Mål med features.
Iterations: Tidsbestemt.

H2 = nytt kapittel.

Idé:
Trykke i teksten og hoppe til plass i lyden.

“Tagge” prosjekter/opptak.

Prosjektforløp med møter.

Essens:

Få ut essensen uten å måtte “lete” etter spesifikke ting. Få en listen/sammendrag, kunne ta vekk ting som ikke er viktige.

Møte om prototyper med Kodebyraaet 08.02.21

Første prototype:

- Trenger mer sjel og “nice” features før kommersialisering og bruk.

Andre prototype:

- Ikke redigeringsknapp
- Timestamps trenger kanskje ikke være med
- Timestamps med link
- Direkte redigering

Tredje:

- Tid til høyre for knapp
- Lenge det varer helt til høyre
- Overkommelig MVP
- Tilføre “nices”.

Generelt:

- Landingsside / Get started
- Hva er det? Hva skal det bli?
- Viewen: Se Dropbox paper for inspirasjon
  - Hele flaten handler om teksten
  - Åpne det opp som basic teksteditor

- Mer innbydende til invitasjon

- Knappene er mer innbydende nå
- Liker to felt. Ett for transkribering og ett for redigering
- Liker idéen om “tags”
- Enklere side med summary/referat som hovedfunksjon.
- Kan få hjelp av KB til hvordan vi kan løse ting. F.eks. tags.

- FLaten hvor deltakere møtes i etterkant?
- Passe på at det ikke blir for stort i starten.
- Er det viktigste transkribering?

Dele:

- Utgangspunktet
- Viktigste: Lydfil + sammendrag

Nøkkel-funksjonalitet:

Møte -> Hente ut informasjonen enkelt og effektivt

Dele med unik URL.

States:

1: laste opp

2: transkribere, prosjektview: lyd, transkribering, lage kapittel?

3: med-deltaker se og lese referatet samtidig. !Lage oppsummering!

- Legge til filer i tillegg. PPT, docx
- Meeting manageR?
- Hente ut og distribuere dokumenter fra møter umiddelbart

Prioriteringsliste:

1. !video
2. !sammendrag

3. tidsstempel for navigering?
4. vedlegg

- Hashet URL (med passord) for å åpne. Ikke nødvendig for MVP.

Minimum:
laste opp webm
view med video, transkribering, summary
distribuere med hashet url (mulig for avansert for oss)

Mange idéer underveis. Noen mer realistiske enn andre. Prøver å holde oss til “scope”.

Møte med Kodebyraaet 08.03.21

!Veldig positivt!
Eliminerer skriving.
Skrubbe i teksten -> finne ut hvor man er i transkriberingen.
Timestamps.
Forbedre output. Tydligere hvem som snakker/hvem som er hvem.

Identifisere folk. Fylle inn navn og deligere til "speakers". Fargekode setninger.
Tenke på deling, formatere og legge inn der man typisk deler slike ting.
Dele til -> Slack, Email, Dropbox Paper.
Dele nettsiden?

Få til speakers. Linjeskift på speakers.
Outputen blir lagt rett inn i texteditor.
Laste opp og velge språk, eventuelt oppdage språk.
Dritbra, imponert, keep it going!

Prioritering: Speakers -> Timestamps -> Deling.

Torsdag -> 15-15.30

Tilbake til teksten.

Vedlegg C: Personas

Alex
Tech-lead i en startup

33 år
Bor i Oslo

Behov
Har mange møter hver dag, sliter med å få notert fra alle møter og følge med 100%. Trenger et verktøy som gjør dette for seg. Behøver at verktøyet tar i mot lyd og videofiler slik han ikke trenger å konvertere.

Frank
Lærer på videregående

53 år
Bor i Ålesund

Behov
Har en elev med nedsatt hørsel, og vil derfor gi h*n mulighet til å lese et transkript av forelesningen, om h*n ikke har fått med seg alt. Trenger også et tekstredigerings-verktøy for å formatere teksten.

Linda
Forsker

41 år
Bor i Oslo

Behov
Har mange lange intervjuer. Trenger en måte å raskt finne i tidslinjen hvor ting ble sagt. Vil finskrive intervjuene og plukke ut de viktigste punktene.

Katarina
Elev

13 år
Bor i Trondheim

Behov
Har akkurat flyttet til Norge og har manglende norskkunnskaper. Deltar i digital undervisning på skolen og har opptak. Kan transkribere opptakene og bruke Google Translate til å oversette til sitt språk.

Bente
Pensjonert barnehagelærer

69 år
Bor i Bodø

Behov
Har nedsatt hørsel etter et arbeidsliv i barnehage. Hun er i karantene og hun trenger transkribering av videomøter med legen sin.

Kurt
Student

21 år
Bor i Kristiansand

Behov
Har flyttet alene for å studere. Klarer aldri å stå opp til digital forelesning. Utsetter alltid å lese til siste liten. Trenger å transkribrere forelesninger slik at han kan skumme gjennom før eksamen.

Tilbake til teksten her

Vedlegg D: Lenke til Miro

Lenke til hele Miro-tavlen fra GDS: https://miro.com/app/board/o9J_lZUKoDU=/

Her ligger GDS-resultatene av mobil- og web-versjonene.

Tilbake til teksten her.

Vedlegg E: Brukertester

Brukertest 1

Pre-test:

Kan du fortelle litt om deg selv?

Dame, 29

[Probe] Hva er yrket ditt?

Lege

Hvor ofte transkriberer du video eller lyd til tekst?

Sjelden.

[Probe] Når gjorde du dette sist?

Hver dag?

[Probe] Bruker du noen verktøy til å gjøre dette? I så fall, hvilke?

Max Manus

[Probe] Kan du beskrive opplevelsen din med det verktøyet?

Veldig variert. Av og til bra. Av og til dårlig.

Test:

Scenario 1:

1. Du åpner nettsiden og laster opp et opptak
2. Opptaket ditt er transkribert. Du vil dele det med resten av deltakerne fra møtet.

Scenario 2:

1. Du ønsker å kopiere den transkriberte teksten.

Hva tenker du når du prøver å få opptaket ditt transkribert?

Det ser bra ut.

Hvordan opplevde du å bruke produktet til å gjøre denne oppgaven?

Veldig bra.

[Probe] Hvor enkelt eller vanskelig var det å navigere produktet?

Faktisk veldig enkelt.

[Probe] Hva tenker du om designet og layouten av produktet?

Fint.

[Intern] Hvor lang tid tok det å løse oppgaven?

Kort.

Motivasjon:

Hvorfor [spesifikk handling som ble gjort]?

-

Post-test:

Kan du beskrive den overordnede opplevelsen din med produktet?

Brukervennlig.

Hva likte du mest med produktet?

At den gjorde det den ville

Hva likte du minst?

Ikke fikk lov å skrive melding

Var det noe som overrasket deg?

At vi snakket spansk

Var det noe som frustrerte deg?

Nei

Var noe forvirrende, eller uoversiktlig?

Nei. Få ting å trykke på og det er bra.

Hvordan synes du det gikk å utføre de oppgavene du fikk?

Det var enkelt for det var få ting å gå seg vill I.

Hva synes du om den generelle utformingen og navigeringen?

Bra.

Følte du noe var unødvendig, eller i veien?

Nei.

Er det noe du savnet ved å transkribere opptakene dine på denne måten?

Usikker på hvem som har sagt hva?

På en skala fra 1-5, hvor stor sjans er det for at du hadde anbefalt dette produktet til en venn?

5?

Hvor ofte ville du brukt dette produktet?

Hver gang jeg har møte.

Hva hadde vært din motivasjon for å bruke et slikt produkt?

Å slippe å transkribere og/eller ta notater underveis.

Avsluttende tanker:

Copy gjør det hun forventer.

Hadde foretrukket fullskjerm.

Fullskjerm:

Her kan jeg redigere? (ser ikke slik ut på liten versjon)

Er det lov å redigere?

Deiligere og mer oversiktlig.

Hva gjør brukeren?

- Trykker select file, velger filen.

- Transkriberer

- Trykker share. Prøver å skrive melding.

- Trykker share igjen og deler med de to

- Scroller litt i transkriberingen.

Brukertest 2

Pre-test:

Kan du fortelle litt om deg selv?

29. Arbeider i barneverntjeneste. Master i sosialt arbeid. Masteroppgave om minoritetsforeldre i Norge. Liker yoga.

Hvor ofte er du i møter:

Jobber 50%. Er i møte hver dag.

Noterer noen møtene:

Ja

Noterer du møtene?

Ofte.

Har 2-3 møter om dagen. Noterer alle.

[Probe] Hva er yrket ditt?

-

Hvor ofte transkriberer du video eller lyd til tekst?

På generell basis kanskje tre-fire ganger i året på arbeid.

Når jeg har skrevet master: Fem ganger på to måneder.

[Probe] Når gjorde du dette sist?

-

[Probe] Bruker du noen verktøy til å gjøre dette? I så fall, hvilke?

Nei

[Probe] Kan du beskrive opplevelsen din med det verktøyet?

-

Test:

Scenario 1:

1. Du åpner nettsiden og laster opp et opptak
2. Opptaket ditt er transkribert. Du vil dele det med resten av deltakerne fra møtet.

Hvordan opplevde du å bruke produktet til å gjøre denne oppgaven?

Veldig lettvint.

[Probe] Hvor enkelt eller vanskelig var det å navigere produktet?

Enkelt

[Probe] Hva tenker du om designet og layouten av produktet?

Heilt kurant, enkelt design.

[Intern] Hvor lang tid tok det å løse oppgaven?

Kort.

Motivasjon:

Hvorfor [spesifikk handling som ble gjort]?

Scrolle på sida.

Scenario 2:

1. Du ønsker å lytte til transkriberingen
2. Du hører at transkriberingen av opptaket ikke er 100% korrekt.
   Dermed ønsker du å kopiere teksten slik at du kan redigere den selv.

Hva tenker du når du prøver å gjøre disse oppgavene?

Enkelt å følge stegene.

Hvordan opplevde du å bruke produktet til å gjøre disse oppgavene?

Bra

[Probe] Hvor enkelt eller vanskelig var det å navigere produktet?

Enkelt.

[Probe] Hva tenker du om designet og layouten av produktet?

Kurant og enkelt, men kjedelig med svart-hvitt. Liker automatikken i det. Andre produkter jeg har brukt før har vært mer avanserte.

[Intern] Hvor lang tid tok det å løse oppgaven?

-

Motivasjon:

Hvorfor [spesifikk handling som ble gjort]?

Post-test:

Kan du beskrive den overordnede opplevelsen din med produktet?

Raskt og lettvint

Hva likte du mest med produktet?

At man slipper å skrive et ord

Hva likte du minst?

At alt er svart og hvitt utenom share-knappen

Var det noe som overrasket deg?

At scrolleknappen ikke fungerte (figma)

Var det noe som frustrerte deg?

Scrolleknappen

Var noe forvirrende, eller uoversiktlig?

Nei

Hvordan synes du det gikk å utføre de oppgavene du fikk?

Greit

Hva synes du om den generelle utformingen og navigeringen?

God

Følte du noe var unødvendig, eller i veien?

Nei

Er det noe du savnet ved å transkribere opptakene dine på denne måten?

Skulle vært tidslinje under avspilleren.

Jeg vil se hva som står på xx:xx vil jeg trykke på tidslinje.

På en skala fra 1-5, hvor stor sjans er det for at du hadde anbefalt dette produktet til en venn?

5!

Hvor ofte ville du brukt dette produktet?

Hver gang jeg transkriberte.

Hva hadde vært din motivasjon for å bruke et slikt produkt?

At man slipper å sitte og skrive hvert eneste ord man hører.

Noen ganger når jeg transkriberer i jobbsammenheng bruker jeg ofte fire timer på et opptak som varer i en time. Dette ville kortet ned tiden det tar pluss at jeg kunne gjort andre ting samtidig.

Fullskjerm:

-

Avsluttende tanker:

Veldig lettvint.

Vanskelig å laste ned andre programmer for transkribering under masteroppgave. Lot det være og gjorde det selv manuelt.

Vil du heller ta opp møter og bruke dette verktøyet?

Ikke i fysiske møter.

Hva gjør brukeren?

-Select file

-Fil

-Transcribe

-Scroller i tekst

-Share

-Message

-Share

-Scroll

-Klikker på scrollbar

-Play

-Pause

-Scroll

-Copy

Brukertest 3

Pre-test:

Deltar du ofte er du i møter?

Ja, akkurat nå digitale

Hvis ja. Er de digitale eller fysiske?

ja

Noterer noen møtene?

noen ganger

Noterer du

om hun har møte

Kan du fortelle litt om deg selv?

Er 30, kvinne,

[Probe] Hva er yrket ditt?

Jobber som web-koordinator for flyktninghjelpen, for tiden på hjemmekontor

Hvor ofte transkriberer du video eller lyd til tekst?

Aldri

Test:

Scenario 1:

1. Du åpner nettsiden og laster opp et opptak
2. Opptaket ditt er transkribert. Du vil dele det med resten av deltakerne fra møtet.

Hvordan opplevde du å bruke produktet til å gjøre denne oppgaven?

Gikk veldig enkelt, men stoler ikke på at det er gjort riktig. Kan jeg sjekke det? endre det?

[Probe] Hvor enkelt eller vanskelig var det å navigere produktet?

Tenkte ikke, veldig intuitivt

[Probe] Hva tenker du om designet og layouten av produktet?

- Litt kjedelig kanskje. Ser ikke helt ferdig ut på en måte. Veldig oversiktlig. Er ting lagret på nettsiden? Kan jeg finne et slags arkiv med transkriberingene mine?

[Intern] Hvor lang tid tok det å løse oppgaven?

-

Motivasjon:

Hvorfor [spesifikk handling som ble gjort]?

Scenario 2:

1. Du ønsker å lytte til transkriberingen
2. Du hører at transkriberingen av opptaket ikke er 100% korrekt.
   Dermed ønsker du å kopiere teksten slik at du kan redigere den selv.

Hva tenker du når du prøver å gjøre disse oppgavene?

- Det gikk veldig fint det. men dumt at man ikke bare kan redigere det her inne. Så hvis jeg klipper det herfra skjer jo endringene bare i word. Hva får de om jeg deler dette? Får de det som en lenke eller som en tekst? Hva om alle bare kunne fått en lenke så, så kan alle redigere og finne/lagre det her?

Hvordan opplevde du å bruke produktet til å gjøre disse oppgavene?

- Virker greit, savner arkiv, kanskje video? Home er rett på forsiden, er det ikke noe om dere? Bør være noe om siden og produktet

[Probe] Hvor enkelt eller vanskelig var det å navigere produktet?

- Veldig enkelt

[Probe] Hva tenker du om designet og layouten av produktet?

- Knappen med select file, endrer teksten til "transcribe".. Bør ikke det være en annen knapp?

[Intern] Hvor lang tid tok det å løse oppgaven?

-

Motivasjon:

Hvorfor [spesifikk handling som ble gjort]?

-

Post-test:

Kan du beskrive den overordnede opplevelsen din med produktet?

- God opplevelse ..ja. eeeh, men jeg skjønner ikke hva jeg skal gjøre med dette, litt irriterende at jeg må lagre det et annet sted. Hvordan kommer tidsstemplene?

Hva likte du mest med produktet?

- At det er enkelt å skjønne hva man skal gjøre, fordi det ikke er så mye dilldall, som er en litt motsetning til hva jeg sa istad. Men

Hva likte du minst?

- At jeg ikke vet helt hva jeg skal gjøre med dette nå. At jeg må ha et annet system for å lagre det

Var det noe som overrasket deg?

- Nei, ikke egentlig. Disse tidsstemplene var veldig smarte!

Var det noe som frustrerte deg?

- Nei

Var noe forvirrende, eller uoversiktlig?

- Nei

Hvordan synes du det gikk å utføre de oppgavene du fikk?

- Det gikk bra!

Hva synes du om den generelle utformingen og navigeringen?

- Litt kjedelig på forsiden. Savner litt info. Veldig sort og deprimerende, vil gjerne trykke på "upload"-ikonet.. vil også trykke på "x". Det er ikke noe problem med navigeringen nå.

Følte du noe var unødvendig, eller i veien?

- nei, bortsett fra ikonet

Er det noe du savnet ved å transkribere opptakene dine på denne måten?

- Ja, arkiv og man kan redigere på websiden. Kunne sende link med transkriberingen.

På en skala fra 1-5, hvor stor sjans er det for at du hadde anbefalt dette produktet til en venn?

- Kan kanskje anbefale det om noen er ute etter en slik løsning: 3-4 kanskje

Hvor ofte ville du brukt dette produktet?

- Ikke ofte. Har ikke behovet

Hva hadde vært din motivasjon for å bruke et slikt produkt?

- kan det brukes til notater kanskje?

- Tror ikke jeg selv kan ha nytte av det, men for eksempel når flyktninghjelpen har styremøte, er behovet større for presise referater. Der kunne det vært et godt verktøy.

Avsluttende tanker:

Føler jeg har sagt det jeg har tenkt underveis..

Hva gjør brukeren?

- Trykker select
- Velger lydfil
- Ser at den ligger der.
- Trykker transkriber
    - Liker at den jobber

- Trykker share
- Prøver å legge til fler
- Prøver å skrive melding.
- Trykker share!
- Sier hurra!

- Scroller i teksten.
- Trykker play og pauser flere ganger.

- Trykker copy.

- Trykker hjem!
- Prøver å trykke på "drag & drop recording"
- Trykker tusen ganger på teksten der det står "upload"
- Går igjennom hele prosessen en gang til
- Trykker på timestamps

Tilbake til teksten her.

Vedlegg F: Om Scrum

Utvikling med Scrum blir delt opp i såkalte “sprinter”. Sprinter er perioder på opp til en måned. De slutter på et bestemt tidspunkt, uavhengig om arbeidet er ferdig eller ikke. I starten av en sprint velger teamet oppgavene som skal gjøres basert på en prioriteringsliste. Disse endres ikke underveis. Teamet møtes hver dag for å oppdatere hverandre på fremgangen. Dette kalles “daily scrum”. På slutten av en sprint samles teamet og interessentene/kunden. Her demonstreres det som har blitt gjort og det blir utvekslet tilbakemeldinger som kan tas med i neste sprint. I motsetning til en vannfallsmetode hvor man planlegger veldig detaljert i begynnelsen før man utfører prosjektet steg for steg ligger Scrums styrke i å tilpasse seg hele tiden. Korte, iterative perioder med planlegging, utvikling, tilpasning gjør at man gradvis kommer nærmere en god løsning på en fleksibel måte.

Scrum har også et sett med roller som skal tildeles. Disse er Product Owner (herfra kalt PO), teamet og Scrum Master (herfra kalt SM). Sammen er disse Scrum Teamet. PO har ansvaret for å maksimere avkastning på investering. Med dette innebærer å identifisere og prioritere funksjoner, og kontinuerlig bestemme hvilke funksjoner som er viktigst å fokusere på til neste sprint. I tilfelle løsningen skal være intern og ikke nødvendigvis har som mål å tjene penger, er det fortsatt POs oppgave å velge den funksjonaliteten med størst verdi som skal fokuseres på.

Teamet sin oppgave er å utvikle produktet som PO ønsker. For eksempel en nettside. Teamet består som regel av ca. syv personer. Avhengig av løsningen som skal utvikles kan det bestå

av folk med egenskaper innenfor bl.a. analyse, utvikling, testing, design og dokumentering. Teamet utvikler produktet og gir tilbakemeldinger til PO om hvordan det kan bli bedre. SMs rolle er å hjelpe resten av teamet med å anvende Scrum og gjør det som trengs for at teamet og PO skal være framgangsrike. Likevel er ikke SM en prosjektleder. SM skal bistå ved å beskytte teamet fra forstyrrelser utenfra og å lede teamet i riktig retning slik at de bruker Scrum riktig (Sutherland og Schwaber, 2007, s. 14-18).

Tilbake til teksten.

Vedlegg G: Om Kanban

Kanban er en agil metodikk som bidrar til å danne et bilde av flyten og arbeidets helhet ved å plassere oppgaver på et såkalt Kanban-brett. I motsetning til Scrum bruker ikke Kanban noen form for rolleinndeling eller iterasjons-faser. Kanban baserer seg mer på visualisering av arbeidet som må gjøres, hva som gjøres nå og hva som er gjort. Ved å begrense arbeidsmengden til mindre oppgaver blir det lettere å prioritere oppgaver og unngå at de akkumulerer seg (Granulo og Tanovic, 2019, s. 1).

Tilbake til teksten.

Vedlegg H: Fremdriftsplaner

Fremdriftsplan 1: Før oppstart.

På anbefaling fra intern veileder: To ganger sprint/research, prototyping, utvikling.
Vi er usikre på om vi har tid til dette.

Uke	Hva	Resultat (output)	Kommentar

2	Research andres løsninger, skrive prosjektbeskrivelse, idemyldring	Finne den beste løsningen på vår problemstilling	
3	Google Design Sprint 1	Grunnleggende prototype.	Ikke design, men brukerproblem og løsninger
4	Google Design Sprint 2	Grunnleggende prototype 2.	Ikke design, men brukerproblem og løsninger
5	Prototyping. Brukertesting, undersøkelser, feedback	Klikkbar prototype. User stories for mvp v1. Bedre prototype/iterasjon..	Design og visuelle profiler. Iterering av versjoner.
6	Iterere. Skrive brukerhistorier.	Oversikt over funksjoner. Ferdig prototype	
7	Research teknologi, rammeverk, bibliotek osv.	Kunnskap før påbegynt utvikling.	
8-10	Utvikle første prototype.	Grunnleggende	MVP
11	Google Design Sprint 3	Grunnleggende prototype 2.	Ikke design, men brukerproblem og løsninger
12	Prototyping. Brukertesting, undersøkelser, feedback	Klikkbar prototype. User stories for mvp v1. Bedre prototype/iterasjon..	Design og visuelle profiler. Iterering av versjoner.

13	Iterere. Skrive brukerhistorier.	Oversikt over funksjoner. Ferdig prototype	
14-18	Utvikle resten av løsning.		Innlevering
18 - 20	Rapportskrivning		Innlevering

Fremdriftsplan 2: Underveis i prosjektet.

Uke	**Hva**	**Resultat (output)**	**Kommentar**
2	Research andres løsninger, skrive prosjektbeskrivelse, idemyldring	Finne den beste løsningen på vår problemstilling	
3	Google Design Sprint 1: mobilapplikasjon	Grunnleggende prototype.	Ikke design, men brukerproblem og løsninger
4	Google Design Sprint 2: nettbasert verktøy	Grunnleggende prototype 2.	Ikke design, men brukerproblem og løsninger
5	Prototyping. Brukertesting, undersøkelser, feedback	Klikkbar prototype. User stories for mvp v1. Bedre prototype/iterasjon..	Design og visuelle profiler. Iterering av versjoner.

6	Iterere. Skrive brukerhistorier.	Oversikt over funksjoner. Ferdig prototype	
7-8	Research teknologi, rammeverk, bibliotek osv.	Innsikt før påbegynt utvikling.	
8-10	Start på utvikling av fundament	Fungerende prototype	
11	Iterering. Mer research før videreutvikling.	Ytterligere innsikt, etter påbegynt utvikling.	
12-13	*Eksamen i utviklingsmetoder.* Brukertester laste-animasjon.	Kunnskap om laste-animasjoner.	
14-18	Utvikle resten av løsning.	Fullført MVP.	
19 - 22	Rapportskrivning	Fullført bachelorprosjekt	Innlevering

Tilbake til teksten.

Vedlegg I: Om React Hooks

“En hook er en funksjon som gir tilgang til React-features, som tidligere kun var tilgjengelige i klassekomponenter, i funksjonskomponenter. De gir funksjonskomponentene dine superkrefter, og muliggjør deling av funksjonalitet på helt nye og spennende måter.”(Selbekk and Gjøby, 2021).

Tilbake til teksten.