Høgskulen på Vestlandet

BACHELOROPPGAVE

Geppetto – fra Windows til Web

Geppetto – from Windows to the Web

Håkon Grimen, Håkon M. T. Markhus, Markus L. Gustavsen

Dataingeniør og informasjonsteknologi
Fakultet for ingeniør- og naturvitenskap
Veileder Richard Kjepso
21.05.2023

Jeg bekrefter at arbeidet er selvstendig utarbeidet, og at referanser/kildehenvisninger til alle kilder som er brukt i arbeidet er oppgitt, *jf. Forskrift om studium og eksamen ved Høgskulen på Vestlandet, § 10.*

Høgskulen på Vestlandet

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

TITTELSIDE FOR HOVEDPROSJEKT

Rapportens tittel:	*Dato:*
Geppetto – fra Windows til Web	*21.05.23*
Forfatter(e):	*Antall sider u/vedlegg: 51*
Håkon Grimen, Håkon Markhus og Markus Gustavsen	*Antall sider vedlegg: 97*
Studieretning:	*Antall disketter/CD-er:*
Dataingeniør og informasjonsteknologi	0
Kontaktperson ved studieretning:	*Gradering:*
Richard Kjepso	Ingen
Merknader:	
Ingen	

Oppdragsgiver:	*Oppdragsgivers referanse:*
Vizrt	EB-14
Oppdragsgivers kontaktperson:	*Telefon:*
Roger Sætereng	922 15 303

Sammendrag:

Prosjektet har som mål å utvikle en web-applikasjon som skal kontrollere en medie-sending. Applikasjonen skal fungere som et supplement til et eksisterende program for Windows, og kommunisere med dette. Dette vil gjøre det mulig å styre en sending fra andre enheter enn tidligere, og da gjøre det mer tilgjengelig. Eksempelvis vil en kunne styre en sending fra mobilen, mens en sitter foran kamera.

The project aims to develop a web application that will control a media broadcast. The application is intended to act as a supplement to an existing program for Windows, and communicate with it. This will make it possible to control a broadcast from devices other than those previously used, thereby making it more accessible. For example, one could control a broadcast from a mobile phone while sitting in front of the camera.

Stikkord:

Webapplikasjon	Kontrollgrensesnitt	Vue.js og TypeScript

FORORD

Takk til oppdragsgiver for en spennende oppgave med fine utfordringer. Gruppen har fått utfolde seg mye og fått god hjelp der det trengs. Gruppen ønsker også å takke for tilretteleggingen vår oppdragsgiver har gjort, dette både med tilgjengelighet av kontorplasser, integreringen i bedriften og den store friheten til å utforske gjennom en ganske åpen oppgave.

Veilederen for gruppen har gitt gode tips om hvordan en god rapport skal utformes både med tanke på utseende og innhold, hva som trengs for å stå løpet ut og generelt vært en svært nyttig ressurs.

INNHOLDSFORTEGNELSE

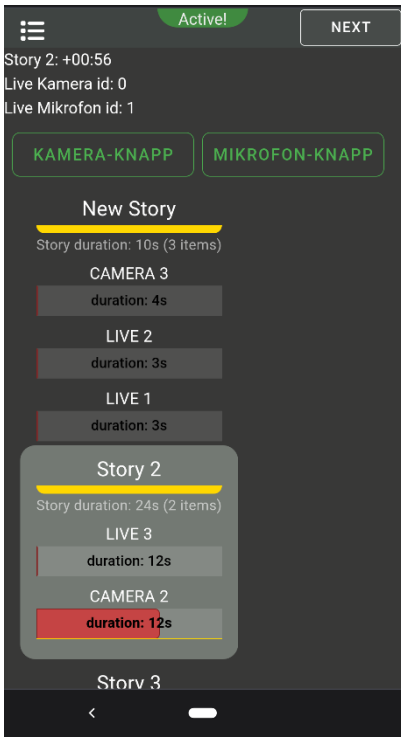
1 INNLEDNING

Denne oppgaven tar utgangspunkt i et program som eies av Vizrt AS, og som brukes til å styre mediasendinger, da i hovedsak direktesendte TV-sendinger. Dette er for øyeblikket et Windows-program der brukeren både trenger opplæring og potensielt hjelp til installasjon. Gruppens prosjekt er å lage en web-basert kontroll-applikasjon som kan kommunisere direkte med backenden til det originale programmet, Viz Mosart (Vizrt Group, 2022).

Dagens teknologiske konkurransemiljø er i stadig endring, og en utvidelse av tjenestene en tilbyr kan ha stor effekt på antallet kunder en kan nå ut til. Ved å utvikle en enkel web-applikasjon vil en senke det nødvendige kompetansenivået som kreves for å benytte seg av deres systemer, og dermed ha muligheten til å innta et betraktelig større marked enn det som var mulig tidligere.

1.1 Kontekst

I dagens samfunn er det en økende etterspørsel etter digitale medieplattformer og innhold som er tilgjengelig på nett. Gruppens prosjekt følger en utvikling i samfunnet, hvor en stadig større andel foretrekker å bruke mindre enheter (Statista, 2023). Som følge av dette vil prosjektet ha et stort fokus på kompatibilitet på tvers av enheter og plattformer.



Figur 1.1 Webapplikasjonen på mobil

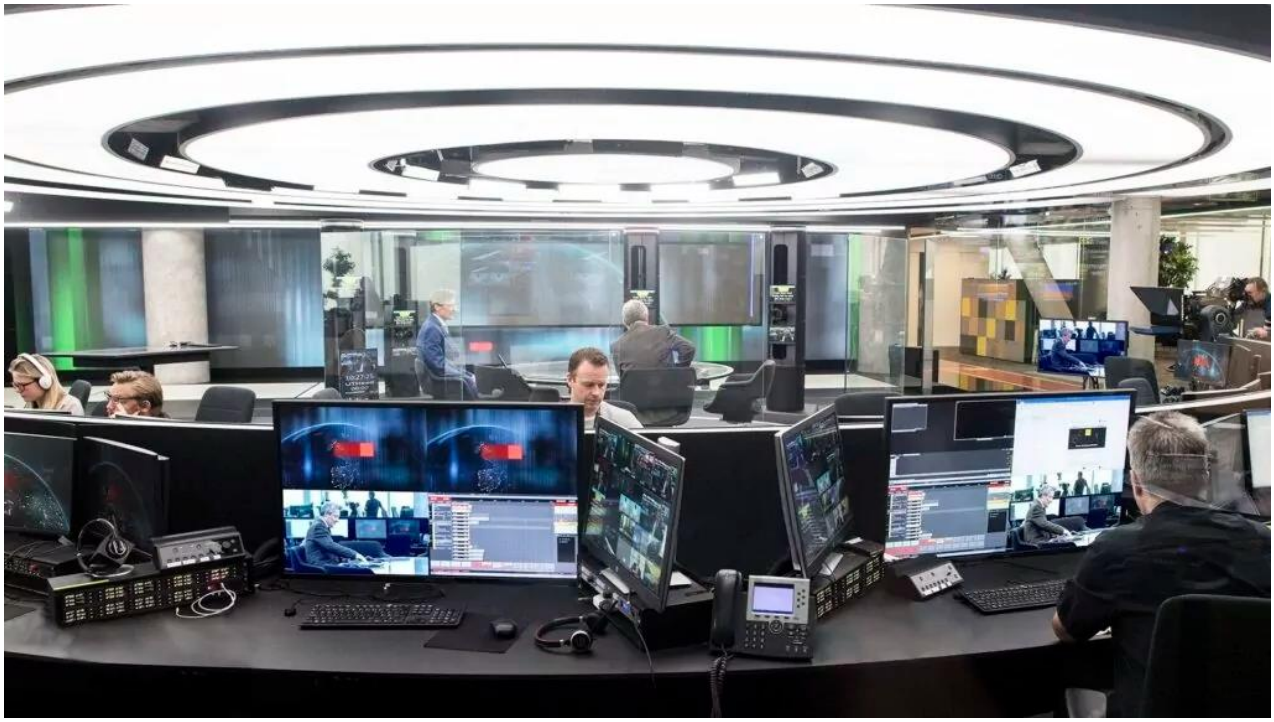
Vårt prosjekt passer inn i bedriftens ønske om å utvide markedsandelen innenfor et voksende segment av innhold med lavere produksjonskostnad. I takt med at strømming har blitt mer populært, har det og blitt muliggjort for flere enheter. Der man tidligere trengte kraftig utstyr for å produsere innhold, kan dette nå gjøres med enhver mobil enhet med kamera og mikrofon (Meta, 2023). Dette har bidratt til en vekst i både skapere og seere, noe som igjen fører til mer innhold. Fordelene ved å strømme innhold på nett har blitt vektlagt kraftig, og i vaken av strømmebølgen finnes det i dag en vedvarende etterspørsel av verktøy for å optimalisere denne prosessen.

Den største veksten i produsenter av slikt innhold har oppstått i segmentet med lavest budsjett og terskel for publikasjon. Ettersom kompetanse ofte følger de økonomiske kreftene, er det en stor kundegruppe med behov for billigere og enklere løsninger. Vår web-applikasjon vil tilby grunnleggende deler av funksjonaliteten til det eksisterende programmet, men grunnet den forenklede brukeropplevelsen vil den appellere til en betydelig større andel kunder.

1.2 Motivasjon

Hovedmotivasjonen bak prosjektet er å skape en mer tilgjengelig og brukervennlig løsning for selskaper som gjennomfører mediesendinger, ved å utvikle en web-applikasjon som inneholder de nødvendige funksjonene til dette. Målet er å utvikle en kontrollapplikasjon som kommuniserer direkte med det eksisterende backend-systemet, og lar brukeren styre nevnte sendinger fra mer mobile enheter, og med mindre teknisk kompetanse.

Ved å tilby en nettbasert kontrollapplikasjon sikrer vi bred tilgjengelighet til systemet, og lar fremtidige kunder benytte seg av selskapets tjenester fra enhver mobil enhet. På denne måten vil prosjektet være et viktig bidrag til en mer tilgjengelig og brukervennlig tjeneste, uavhengig av teknisk kompetanse, som vil styrke posisjonen til bedriften i markedet.



Figur 1.2 TV-studio. Bildet er hentet fra opppdragsgivers nettside om produktet Viz Mosart. (Vizrt, 2023).

1.3 Prosjekteier

Vizrt AS er et norsk teknologiselskap som spesialiserer seg på løsninger for real-time 3D grafikk, video arbeidsflyt og automatisering for TV- og medieindustrien. Selskapet ønsker å utvikle en web-applikasjon som gjør det enklere og mer overkommelig for TV-stasjoner og mediebedrifter å produsere innhold ved å redusere kostnadene og kompetansebehovet rundt avholdelsen av nevnte mediesendinger.

1.4 Problembeskrivelse og mål

I dag er det skrivebordsprogrammet Viz Mosart som tilbyr styring av medie-sendinger, men programmet krever både installasjon og lengre opplæring. Dette kan være en utfordring, spesielt for mindre omfattende produksjoner.

Målet med denne oppgaven er å utvikle en web-applikasjon som gjør gjennomføringer av TV-sendinger mer tilgjengelig og kostnadseffektivt. En web-applikasjon kan fungere som et supplement til selskapets eksisterende skrivebordsløsning, med kun de mest kritiske funksjonene. Dette vil øke tilgjengeligheten til produktet, og åpne opp for et større marked.

Det første delmålet er å kunne opprette en tilkobling til Mosart, lese inn en sendeplan og gjennomføre den.

Det andre delmålet er å kunne skreddersy sitt eget arbeidsområde, inne i web-applikasjonen.

1.5 Oppbygging av rapporten

Denne rapporten er bygget opp av følgende kapitler:

Tabell 1.1 oversikt over innhold

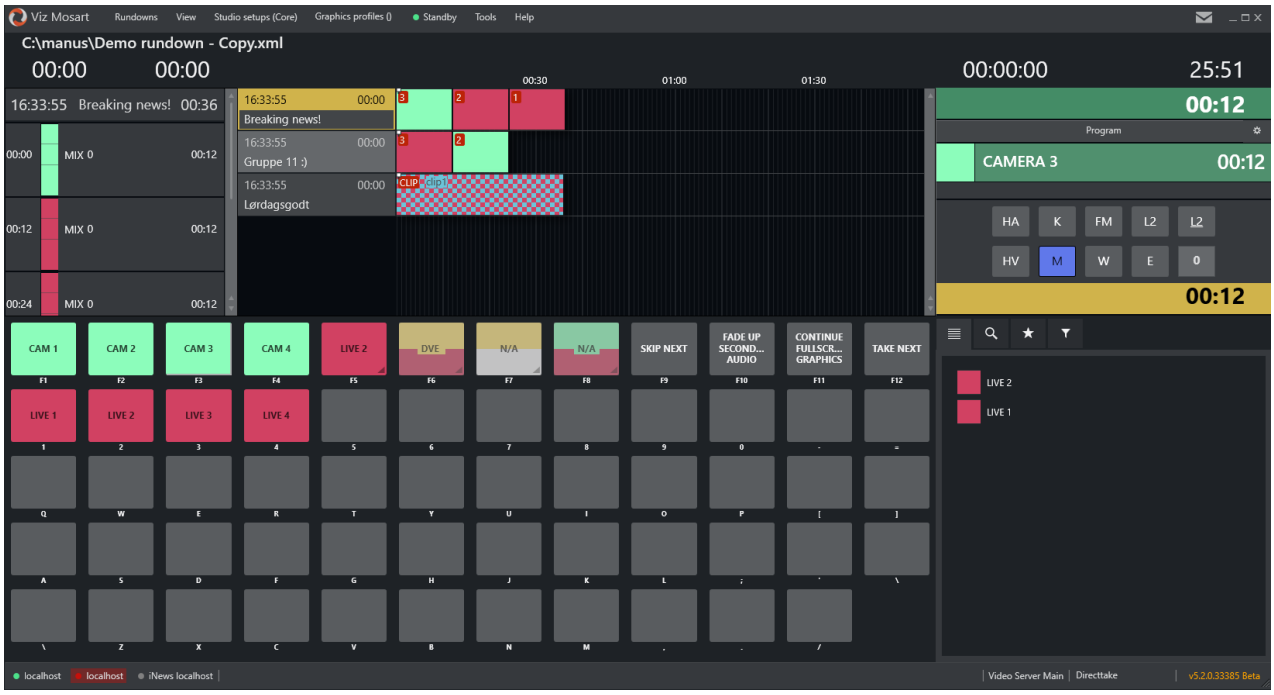
1 Innledning	Motivasjonen bak prosjektet og informasjon om prosjektets eier
2 Prosjektbeskrivelse	Overordnet beskrivelse av prosjektet og dets mål.
3 Design av prosjektet	Forslag til løsning på prosjektet, metodikk og diskusjon rundt hvilke løsninger som ble vurdert.
4 Detaljer løsning	Detaljert beskrivelse av hvordan løsningen er bygget opp, hvilke valg som er gjort og hvordan løsningen ble seende ut.
5 Resultater	Resultatet av prosjektet presenteres og evalueres i detalj
6 Diskusjon	Sluttproduktet, prosessen og gjennomføring av prosjektet diskuteres. Konsekvenser av valgene gjort underveis er også diskutert her.
7 Konklusjon og videre arbeid	Hva som ble oppnådd gjennom prosjektet og hva som kan gjøres videre i et slikt prosjekt
8 Referanser	Inneholder referanser til kilder som er brukt i denne rapporten
9 Vedlegg	Vedlagte støttedokumenter. Dette er prosjekthåndbok, visjonsdokument, kravdokument og systemdokumentasjon.
10 Ordliste	Beskrivelse og forklaringer av de forskjellige ord og begrep som er benyttet i denne rapporten.

2 PROSJEKBESKRIVELSE

2.1 Praktisk bakgrunn

2.1.1 Tidligere arbeid

Prosjektet bygger videre på funksjonaliteten i et program (vist i Figur 2.1) som eies av Vizrt AS. Navnet på programmet er Viz Mosart, og har funksjonaliteten til å kunne styre mediasendinger, slik som TV-sendinger. Denne programvaren gir brukeren muligheten til å sette opp et tidsskjema med kameraer og mikrofoner, for å deretter bruke denne som mal ved direkte strømming. Denne visuelle fremstillingen av en tidslinje er en viktig komponent som ønskes innlemmet i sluttproduktet.



Figur 2.1: Viz Mosart, i desktopversjon. Produkt og bilde tilhører Vizrt.

2.1.2 Initielle krav

De initielle kravene til produktet, er at det skal utvikles en web-basert kontrollapplikasjon som skal kommunisere direkte med Mosart, og la brukeren styre en sending med minimalt av utstyr. De viktigste funksjonene og vår hovedprioritet blir følgende:

- Vise en oversikt over hva som sendes direkte
- En funksjon/knapp som lar en bytte kamera
- En knapp som lar en bytte mikrofon

- En knapp som lar en velge neste, i en predefinert liste
- Muligheten til å designe sitt eget dashbord, med egne knapper (Til flere kameraer, mikrofoner, grafikk, etc.)

Ettersom sluttproduktet skal tilrettelegge for kundene med minst mulig utstyr, har gruppen valgt å fokusere på et design som tilpasser seg etter størrelse, slik at brukeropplevelsen ikke blir svekket på mer mobile enheter.

2.1.3 Initiell løsnings-idé

Oppgavens initiale løsnings-idé er at det skal utvikles en «lettvekts-versjon» av en allerede eksisterende programvare. I ordet lettvekt ligger det her at det er en applikasjon som kjøres i nettleseren, i motsetning til den eksisterende løsningen som er en desktop-applikasjon. Den nye løsning som dette prosjektet omhandler, har et stort fokus på brukervennlighet og et dynamisk design som fungerer like bra på et nettbrett eller mobil, som på en enhet med større skjerm.

2.2 Avgrensninger

Naturlige avgrensninger i dette prosjektet er hovedsakelig å hindre at applikasjonen blir for tung og lite brukervennlig. Med det følger det også en begrensning av tid, som fører til en pragmatisk tilnærming til prioritering av funksjonalitet. Gitt prosjektets mål om å forenkle bruken av selskapets tjenester fokuserer gruppen på grunnleggende funksjonalitet i applikasjonen, og lar heller de mulige tidsmessige begrensninger ramme ønskede ekstra funksjoner.

Oppdragsgivers ønske om at web-applikasjonen skal kunne kjøre på flere typer enheter underbygger dette valget. I dagens samfunn kreves det fleksibilitet i arbeidsmetode, og mobilitet i stadig tilpassede arbeidsforhold. En applikasjon som er like brukervennlig på mobiltelefon som på en stasjonær PC anses som svært viktig og det er noe gruppen har et ønske om å få til. For å sikre at det skjer, må det gjøres nødvendige avgrensninger for å hindre at noe kommer i veien for dette. Funksjoner som kan ses på som nyttige, men ikke nødvendigvis kritiske, kan derfor måtte droppes til fordel for kryss-plattform funksjonaliteten.

2.3 Ressurser

Viktige ressurser for å gjennomføre denne oppgaven finnes i hovedsak hos oppdragsgiver. Ettersom det finnes eksisterende systemer som utfører mye av det samme som vår løsning skal gjøre, er det naturlig å la seg inspirere av denne. Dette gjelder for både design og funksjonalitet.

Teknisk støtte, rådgivning underveis og generell veiledning har gruppen god tilgang på hos oppdragsgiver. Dette, samt tilgang til kontorplass gjennom hele prosjektperioden, er store ressurser som legger til rette for godt samarbeid og kontinuerlig innspill fra oppdragsgiver. Vår veileder ved HVL er en viktig ressurs for gruppen for alt som omhandler rapport, akademisk skriving og gjennomføring en av bacheloroppgave.

Programvaren som brukes for å lage produktet er Microsoft VSCode (Microsoft, 2023) en svært kraftig og konfigurerbar utviklerprogramvare. Dette valget kom etter grundig drøfting, og er basert på en rekke faktorer. Det ble først og fremst anbefalt av utviklere hos oppdragsgiver, og var et verktøy de kunne bistå med mye kompetanse i. En annen viktig årsak bak valget er at det ses på som svært gunstig å benytte seg av VSCode sitt omfattende bibliotek av utvidelser, som blant annet inneholder live-servere for simultan samkjøring under utvikling, og kunstig intelligens som foreslår tilpasninger til koden en skriver.

Valg av utviklerprogramvare er tradisjonelt sett ikke fastlåst innad i et prosjekt, da koding i utgangspunktet bare er redigering av tekstfiler. I tillegg anses ofte erfaring og preferanser som viktigere, kontra samkjøring av utviklerverktøy. Som utviklere har vi ikke store formeninger eller erfaring med spesifikke utviklerverktøy. Den kompetansen gruppen har ligger i flere ulike verktøy som Eclipse og IntelliJ, men også VSCode. Prosjektgruppen har derimot verdsatt et tettere samarbeid med oppdragsgiver, over egen erfaring i utviklerverktøy. Derfor ble valget VSCode.

Figur 2.2 Visual Studio Code logo (Microsoft, 2023)

Etter hvert som prosjektet utvikler vil seg, vil gruppen ha behov for tilgang til endepunkter som programvaren kan kommunisere med. Dette kan være en «demo API» for testing, eller et reelt API som har mulighet til å utføre hendelser basert på kommandoer sendt fra vår applikasjon.

Responsivt design representerer en sentral tilnærming innen moderne webutvikling, hvor målet er å lage nettsider og applikasjoner som tilpasser seg brukerens skjermstørrelse. Gitt oppdragsgivers ønske om en mer tilgjengelig tjeneste er det naturlig at dette har vært høyt prioritert gjennom hele prosjektet. Ønsket om en optimal brukeropplevelse uavhengig av skjermstørrelse har blitt gradvis viktigere i samfunnet, i korrelasjon med økningen i antall mobile enheter (Statista, 2023).

3 DESIGN AV PROSJEKTET

Gruppens begrensede forhåndskunnskap om emnet medførte en nødvendighet av å tilegne seg ny kompetanse kontinuerlig gjennom prosjektets løp. I den innledende fasen fikk gruppen verdifulle innspill fra utviklere hos Vizrt, noe som bidro til å forme beslutningene om valg av teknologi og arbeidsmetoder. På bakgrunn av disse innspillene, samt gruppens egne undersøkelser og ønsker om en intuitiv brukeropplevelse og robust frontend, ble det valgt å utvikle applikasjonen i rammeverket Vue.js.

Vue.js (Vue.js, 2021) er et populært og kraftfullt JavaScript-rammeverk som gir utviklerne en enkel og effektiv måte å bygge interaktive og responsive nettsider på. En web-applikasjon bygget i et reaktivt rammeverk, slik som Vue.js er, kategoriseres som en "Single Page Application" og tilbyr et svært responsivt brukergrensesnitt som sikrer høy brukervennlighet og flyt. Det gir også et stort bibliotek med komponenter og verktøy for å håndtere en rekke oppgaver, inkludert databinding, routing og animasjoner, noe som kan bidra til å øke utviklingshastigheten og effektiviteten. Ved bruk av en «Single Page Application» vil også brukeren slippe å oppdatere web-siden på den tradisjonelle måten ved å laste inn hele siden på nytt, men heller oppleve en rask og dynamisk oppdatert web-applikasjon som henter ny data bare til de delene av websiden som trenger det (Mozilla, 2023).

For å sikre en kodebase som er robust og enkel å vedlikeholde kan det være nyttig å bruke TypeScript som et kodespråk for å utvikle applikasjonen. TypeScript gir utviklerne en høyere grad av sikkerhet ved å validere koden ved kompileringstid, noe som kan redusere antallet feil og forbedre kodekvaliteten. TypeScript støtter også alle funksjoner og funksjoner i JavaScript, slik at utviklerne kan ta fullt utbytte av JavaScripts funksjoner og biblioteker mens de får den ekstra sikkerheten og produktiviteten som TypeScript gir.

3.1 Forslag til løsning

I dette kapittelet presenteres og diskuteres mulige løsninger på problemstillingen gruppen står ovenfor. Drøfting som må gjøres, valg som må tas underveis, argumenter for og mot de forskjellige alternative, og til slutt beskrives det hvilken løsning som ble valgt for dette prosjektet.

3.1.1 Alternativ løsning 1

Et aktuelt alternativ er en løsning der hovedtyngden av funksjonaliteten skjer i selve nettleseren, med størst fokus på frontend. For å bygge slike frontend-løsninger har man mange mulige alternativer av teknologier. Blant de mest populære (Stack Overflow, 2022) er React.js, som i likhet med Vue.js er et JavaScript-rammeverk som tilbyr mange gode og nyttige løsninger som ville passet gruppens prosjekt på en god måte. Et slikt valg ville vært gunstig med tanke på både opplæring og feilsøking, og da spesielt sistnevnte ettersom det er svært mye brukt og har en stor brukerbase. React (Meta, 2023) er bredt støttet og har en stor aktør i ryggen, noe som sikrer langsiktighet og god dokumentasjon.

3.1.2 Alternativ løsning 2

Ettersom oppgaven og bestillingen fra oppdragsgiver er tydelig på at de ønsker en web-basert løsning, er det naturlig å se på de teknologier som brukes i web-applikasjoner. Et av de første spørsmålene gruppen stilte seg selv var om den viktige funksjonaliteten i applikasjonen skulle skje på frontend eller i en backend-løsning.

En backend-basert løsning kunne gitt tilsvarende funksjonalitet som den frontend-løsningen som ble valgt, og det kan argumenteres med at det ville vært en løsning som hadde vært enklere for gruppen å komme i gang med. Dette fordi det gjennom studiet har det vært mye større fokus på programmering og språk om passer for backend, slik som Java, enn det har vært på frontend og web-programmering. Ved å velge en "Single Page Application" bygget i et JavaScript-rammeverk slik som gruppen gjorde, ble læringskurven betydelig brattere enn dersom man hadde tatt mer kjent stoff, men gevinsten og nytten ved å velge et mer progressivt frontend rammeverk anses som stor.

3.1.3 Diskusjon av alternativene

De alternativene som ble vurdert har alle sine fordeler og ulemper. En løsning der det aller meste av funksjonalitet er plassert i frontend vil ofte ha noe lenger innlastingstid, men vil etter det oppleves svært responsiv, rask og fleksibel, noe vi anser som viktig for at denne applikasjonen skal oppleves som god.

En løsning der backend gjør en større del av jobben ville også vært et godt alternativ, men det kan fort oppleves som noe tyngre. Brukervennlighet står høyt og vurderes som svært viktig for å sikre at dette prosjektet resulterer i et produkt som kan brukes videre. En annen årsak til at gruppen ønsket å flytte fokus fra backend til frontend, var at dagens desktop-løsning som brukes flere steder fra før, allerede er tungt forankret i backend. Gruppen tolker oppdragsgivers ønske

dithen at dette er noe de ikke ønsker at en ny løsning skal være, som igjen styrker gruppens valg om å fokusere på frontend.

Ved å velge en løsning som kjører i nettleseren, slik en frontend-løsning gjør, kan gruppen dra nytte av flere av godene et JavaScript-rammeverk tilbyr underveis i utviklingen av en web-applikasjon. Funksjonalitet som «hot-reloading» (Vue.js, n.d.) gjør utviklingsjobben betydelig mer tidseffektiv da en utvikler vil få tilnærmet umiddelbar mulighet til å se hvordan egen kode oppfører seg i nettleseren.

React er et rammeverk som også ble vurdert, primært fordi det er mer populært enn det Vue.js er. Det betyr at det kan være enklere å finne nødvendige ressurser og hjelp, spesielt ved søk på velkjente hjelpesider for koding, slik som StackOverflow.

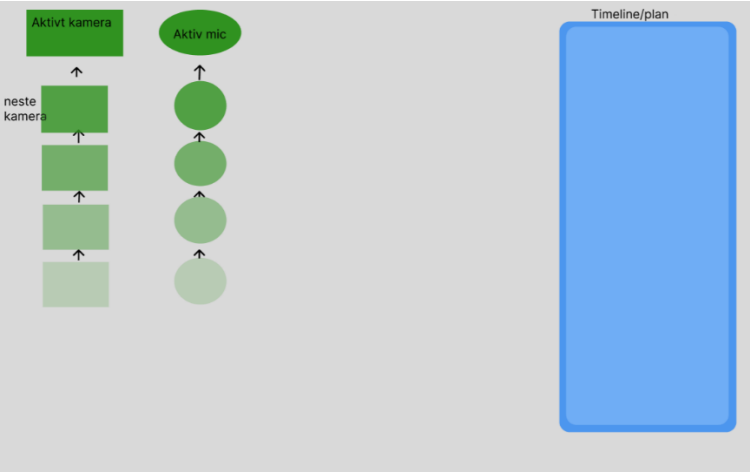
Ved å benytte TypeScript vil det også gjøre at kodebasen er mer robust og sikker. Det kan gjerne innebære en noe tyngre vei til målet for gruppen, men her vurderes det også som et positivt bidrag til helheten gruppen ønsker å oppnå. TypeScript sikrer i tillegg at mulige feil i koden oppdages tidligere enn dersom man hadde benyttet seg av vanlig JavaScript (Microsoft, 2023).

3.2 Valgt løsning

Løsningen som ble valgt av gruppen er en frontend-fokusert løsning som bygges i Vue.js og TypeScript. Dette valget begrunnes på flere måter, men den årsaken som kan anses som viktigst, er at dette er en løsning som er i tråd med det oppdragsgiver bruker internt og har god kunnskap om. Dette betyr at ved å velge Vue.js, har gruppen god support og hjelp tilgjengelig når det behovet oppstår. I tillegg vil en integrering mot eksisterende systemer bli en enklere prosess enn dersom gruppen hadde valgt en annen fremgangsmåte, bygget på andre teknologier.

Det anses også som et fremtidsrettet valg, og det på flere måter. Vue.js er et rammeverk som opplever økende popularitet, og det spås en lys fremtid. Bred støtte, god dokumentasjon og gode utviklingsverktøy gjør at Vue.js er et svært kraftig rammeverk, samtidig som det holder seg allsidig og funksjonelt (Tagline Infotech LLP, 2023).

Dersom oppdragsgiver vurderer produktet som godt nok til å utvikles videre, vil det også være fordelaktig at det er laget i et kjent kodespråk som er i tråd med oppdragsgivers egen filosofi og teknologi-stack.



Figur 3.1 en veldig tidlig skisse. I denne skissen var aktivt kamera og mikrofon mer i fokus enn i en endelig løsning

Gruppen valgte å tegne opp en skisse, som vist i Figur 3.1, av visjonen for applikasjonen. Dette er en tidlig skisse etter funksjonalitet var drøftet og anses som gruppens første utkast og innledende mål. Det tas forbehold om at denne skissen kun er et utgangspunkt, og at sluttresultatet vil påvirkes at prosessen.

3.3 Valg av verktøy

Etter oppstartsmøte med oppdragsgiver ble det drøftet i gruppen hvilke valg vi ønsket å ta om arbeidsverktøy. Her ble det lagt stor vekt på hva dagens utviklere foretrekker, samt hvor kompetansen ligger hos oppdragsgiver, og en beslutning om å bruke VSCode (Microsoft, 2023) og Vue.js 3 (Vue.js, 2021) ble raskt tatt. Dette ble valgt basert på fordelene ved kompetanse i umiddelbar nærhet ved utvikling, i tillegg til et ønske fra studentene om å få et ærligere blikk på fremtidig arbeidsmåte.

Til planlegging og oppgavefordeling har gruppen valgt å bruke Trello (Trello, n.d), et program som lar en definere arbeidsoppgaver og fordele de på gruppens medlemmer. Noen av fordelene med Trello er at en kan markere oppgaver en jobber på, enkelt se hvilke oppgaver som gjenstår, og definere spesifikke instrukser til enhver oppgave. I design-prosessen har gruppen brukt Figma (Figma, n.d.), en nettapplikasjon for design av grensesnitt, for å tegne opp innledende skisser til ferdig produkt.

Ved gjennomføring av større prosjekter er det nødvendig med et robust system for synkronisering av kode, med tilhørende versjonslogg. Til akkurat dette ble gruppen anbefalt Gitlab (Gitlab, 2014) av oppdragsgiver, og fikk deretter en kort introduksjon til dette.

For samhandling og kommunikasjon underveis i prosjekt har gruppen benyttet seg av Google Drive (Google Drive, u. å.) og Discord (Discord, 2023). Dette er to verktøy som forenkler deling av filer, kommunikasjon og generelt samarbeid, i tillegg til at det tilbyr løsninger for fillagring.

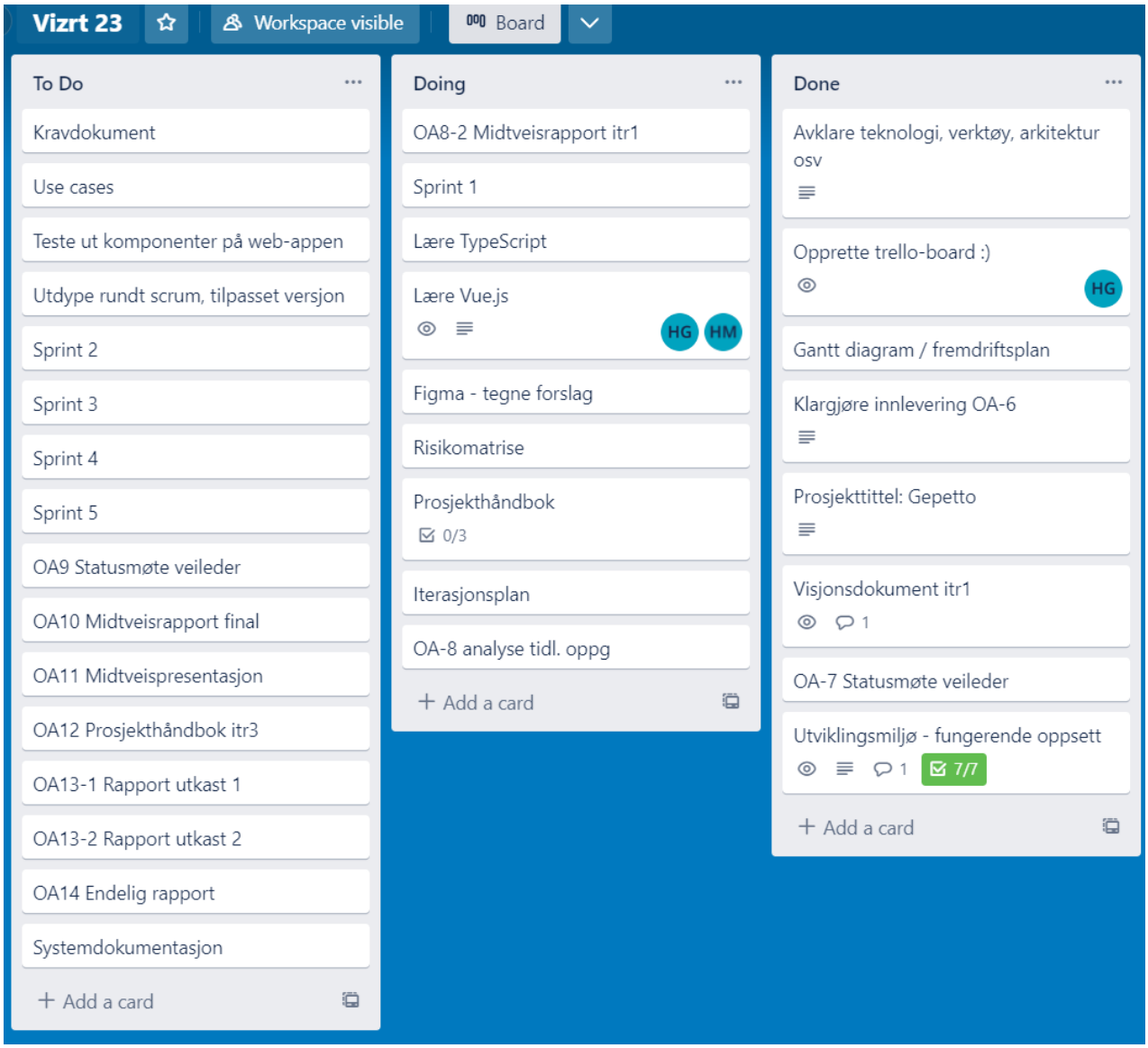
3.4 Prosjektmetodikk

3.4.1 Utviklingsmetodikk

Det ble tidlig tatt et valg om å benytte seg av en agil metodikk (Beck, et al., 2001) for utvikling, støttet av kjente metoder og verktøy fra Scrum og iterasjonsbaserte fremgangsmåter. Scrum (Scrum.org, 2023) er en agil teknikk, utviklet for snart 30 år siden, som legger til rette for korte iterasjoner (også kalt sprinter) med tilpasning og tilbakemelding etter hver sprint. Gruppen har tidligere benyttet seg av den metoden, og finner det tilfredsstillende til prosjekter som dette.

Det ble tidlig satt opp en iterasjonsplan med tydelig definerte delmål som gruppen ønsket å oppnå. En av hovedårsakene bak en iterasjonsbasert metodikk med klare "Minimum Viable Product, MVP", er at det oppleves som motiverende ved gjennomførte delmål og synlig fremgang. Tidlig i et stort prosjekt kan det oppleves som for omfattende og uoppnåelig dersom man ikke har definert mål som kan oppnås relativt tidlig i prosessen.

Innenfor prosjektledelse og utviklingsmetodikk fremstår agil utvikling som et nøkkelkonsept, spesielt i prosjekter som trenger fleksibilitet og kontinuerlig forbedring. Denne tilnærmingen skiller seg mer fra konvensjonelle metodikker, som legger vekt på grundig planlegging og fast struktur. Ved å benytte seg av denne utviklingsmetoden er prosjektgruppen i stand til å håndtere uforutsette situasjoner og tilpasse seg eventuelle endrede ønsker fra oppdragsgiver. Tilnærmingen krever dog en høy grad av samarbeid og kommunikasjon, både med oppdragsgiver og internt i gruppen, for å sikre at prosjektet har størst mulighet for suksess.



Figur 3.2 skjermbilde av gruppens Trello-board fra en tidlig fase i prosjektet.

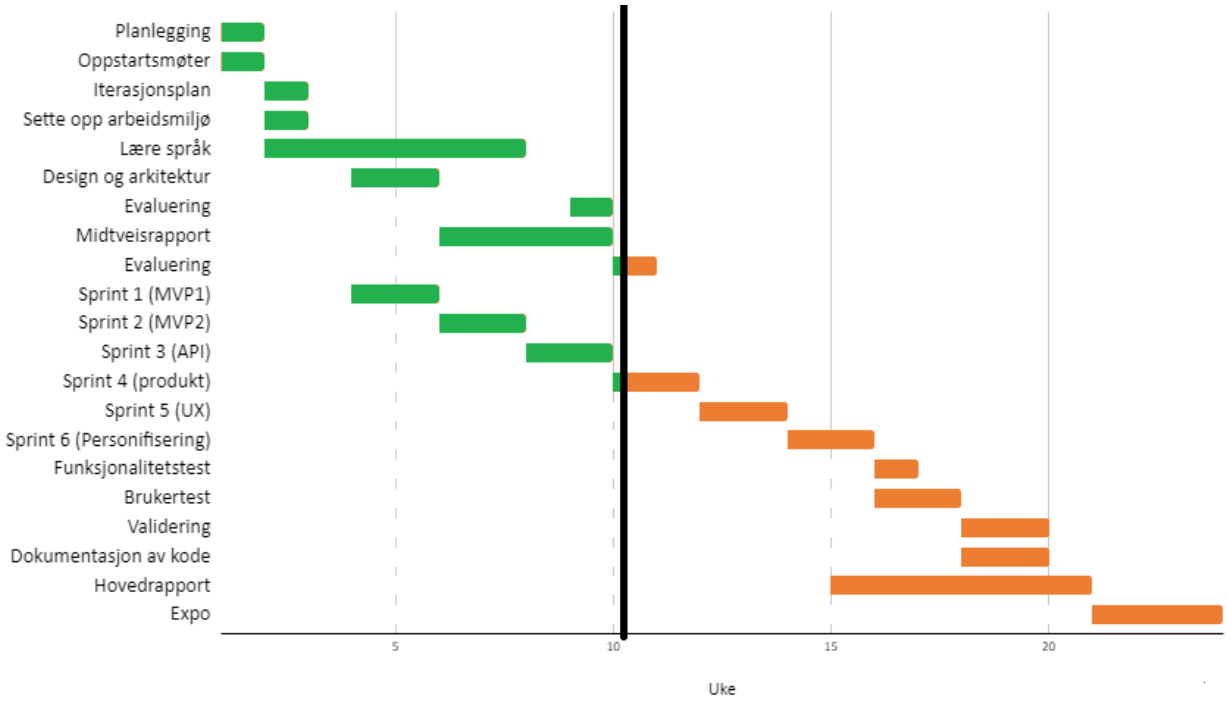
3.4.2 Prosjektplan

Prosjektplanen viser de planlagte aktivitetene og milepælene gruppen har satt opp. Disse er forankret i iterasjonsplanen som tidlig ble definert. Planen vil oppdateres og tilpasses gruppens progresjon etter hvert som tiden skrider frem gjennom prosjektets levetid.

	Oppgave	Start(uke)	Slutt(uke)	Varighet (uke)
Oppstartsfase				
	Planlegging	1	2	1
	Oppstartsmøter	1	2	1
	Iterasjonsplan	2	3	1
	Sette opp arbeidsmiljø	2	3	1
	Lære språk	2	8	6
	Design og arkitektur	4	6	2
	Evaluering	9	10	1
	Midtveisrapport	6	10	4
	Evaluering	10	11	1
Implementering				
	Sprint 1 (MVP1)	4	6	2
	Sprint 2 (MVP2)	6	8	2
	Sprint 3 (API)	8	10	2
	Sprint 4 (produkt)	10	12	2
	Sprint 5 (UX)	12	14	2
	Sprint 6 (Personifisering)	14	16	2
Testing & rapport				
	Funksjonalitetstest	16	17	1
	Brukertest	16	18	2
	Validering	18	20	2
	Dokumentasjon av kode	18	20	2
	Hovedrapport	15	21	6
	Expo	21	24	3

Fargekoder:	
	Planlagt
	Pågår
	Utført

Figur 3.3 Prosjektplan fra prosjektets midtfase



Figur 3.4 diagram tilknyttet prosjektplanen vist over

3.4.3 Risikovurdering

Risikovurdering fra 21.02.23. Dette er en revidert utgave av første versjon, laget tidlig i prosjektets oppstartsfase. Detaljer om endringer, den opprinnelige versjonen og nyere versjoner finnes i Vedlegg 1 Prosjekthåndboken, som følger denne rapporten.

Risikomatrisen som brukes i dette prosjektet:

Sannsynlighet						
	Svært Høy (5)	5	10	15	20	25
	Høy (4)	4	8	12	16	20
	Middels (3)	3	6	9	12	15
	Lav (2)	2	4	6	8	10
	Svært Lav (1)	1	2	3	4	5
		Svært Lav (1)	Lav (2)	Middels (3)	Høy (4)	Svært Høy (5)
	Konsekvens					

Figur 3.5: risikomatrise, hentet fra rapportmal

Tabell 3.1 Risikovurdering, versjon 2

#	Hendelse /Risiko	Årsak	Sannsynlighet	Konsekvens	Risiko-produkt	Tiltak
1	Manglende kunnskap	Studentene mangler kunnskapen til å utføre oppgaven	Svært lav (1)	Høy (4)	4	Lære seg det nødvendige
2	Sykdom	Sykdom	Svært høy (5)	Svært lav (1)	5	Vask og smittevern. Tilrettelagt arbeidssted hjemme
3	Mangelfull eller feilslått planlegging	Fokusere på feil del av oppgaven	Høy (3)	Høy (3)	9	Drøfte valg nøye, gjøre nødvendige endrer i tide. Tett dialog med kunnskapsrike folk hos oppdragsgiver
4	Feiltolking av kravspec/bestilling	Dårlig kommunikasjon, manglende kompetanse	Middels (3)	Høy (4)	12	Bruke tid på kontoret til Vizrt. Sikre god kommunikasjon med oppdragsgiver
5	Feilberegne tidsbruk og ressurser	Vanskelig, undervurdere utfordringer	Høy (4)	Høy (4)	16	Planlegge tidlig, revurdere ofte og sammenligne med erfaringer underveis
6	Overvurdere egen kunnskap	Dönning-krüger effekten	Lav (2)	Middels (3)	6	Ikke ta lett på ting. Være realistisk, bruke tidligere erfaringer og erfaringer underveis til å ta gode og fornuftige valg
7	Dårlig samarbeid internt i gruppen	Uhåndterte konflikter som terger og irriterer.	Svært lav (1)	Høy (4)	4	Lav terskel for diskusjon og hyggelig intensjoner
8	Utstyr/maskinvare som ikke fungerer	Feil bruk, skade under transport osv.	Svært lav (1)	Lav (2)	2	Skylagring av viktige filer og anskaffe nytt utstyr

3.5 Evalueringsplan

3.5.1 Evaluering av webapplikasjonen

Applikasjonen er foreslått testet på kunder hos oppdragsgiver, men det er enda ikke avklart hvorvidt dette blir mulig. Alternativet er brukertesting på fiktive brukere, roller potensielt bemannet av oppdragsgiver eller av gruppens egne medlemmer.

3.5.2 Systemtesting

Systemtesting innebærer at web-applikasjonen testes på hele funksjonaliteten. Det vil testes for ønsket funksjonalitet, opplevelse og generelle oppfattelse av applikasjonen. Målet med testingen er å avdekke potensielle feil eller mangler i koden, slik at korrekte tiltak kan iverksettes. Det erfares at testing av visse komponenter blir mindre nødvendig når en bruker teknologi som oppdaterer det visuelle resultatet dynamisk, da en ofte vil umiddelbart se resultatet av endringen som gjøres. I samråd med oppdragsgiver ble Cypress nevnt som et potensielt verktøy for en automatisert testing av webapplikasjonen, dersom dette ble vurdert som nødvendig.

Brukertesting vil også utføres. Det vil da gjøres av en eller flere personer som bruker applikasjonen i et miljø som simulerer en reell situasjon. Oppdragsgiver vil delta her og komme med sine tilbakemeldinger på helheten og brukeropplevelsen av produktet gruppen har utviklet. Målet med denne testen er å avklare om kravene i oppgaven er innfridd.

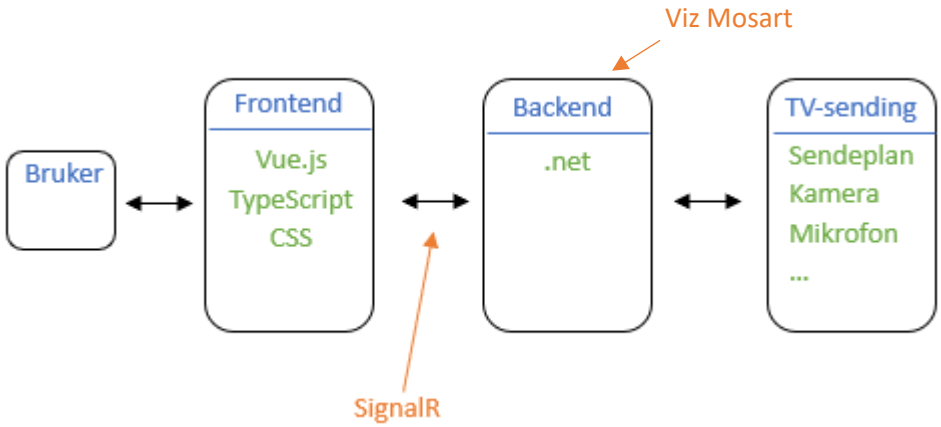
4 DETALJERT LØSNING

Det ble tidlig konkludert med at gruppens hovedfokus skulle være å utvikle en fungerende web-applikasjon, med noen grunnleggende funksjoner. Disse funksjonene skal kommunisere med Vizrt sitt backend system, Viz Mosart, og utføre valgte handlinger i sendingen.

Som et tilskudd rettet mot mobile enheter er applikasjonen begrenset i funksjonalitet, men svært responsiv til ulike skjermstørrelser. I samarbeid med Viz Mosart kan en laste inn en forhåndsbestemt tidslinje, og gjennomføre en slik sending, direkte i nettleseren fra en mindre enhet. Gruppen har valgt å dele opp skjermen inn i ulike soner, hvor hver sone har sin egen funksjon, og hver sone kan tilpasses ut i fra størrelsen og mengden innhold en ønsker synlig. På Figur 4.7 er det et skjermbilde av en tidlig løsning, som viser de ulike sonene i applikasjonen.

Hovedmålet vedrørende designet til applikasjonen har vært å minimere kompetansekrav og opplæringstid, samtidig som en kan tilpasse applikasjonen etter egne behov. På denne måten vil løsningen appellere til kundesegmentet oppdragsgiver ønsket å nå, samtidig som at en har mulighet til å tilrettelegge for mer omfattende produksjoner. Ved å tilby grunnleggende funksjonalitet gjennom en nettleser vil tilgjengeligheten til den nødvendige programvaren øke drastisk.

4.1 Arkitektur



Figur 4.1 arkitekturlagene

Figur 4.1 viser den overordnete arkitekturen og lagene i denne applikasjonen og hvilken teknologi som er brukt på hvert nivå. Boksen merket «TV-sending» kan være en hvilken som helst sending eller studio, som da styres via instruksjoner sendt fra backend. Backend-løsningen er omtalt som Viz Mosart i dette dokumentet, ettersom det er det interne navnet oppdragsgiver bruker for denne funksjonaliteten.

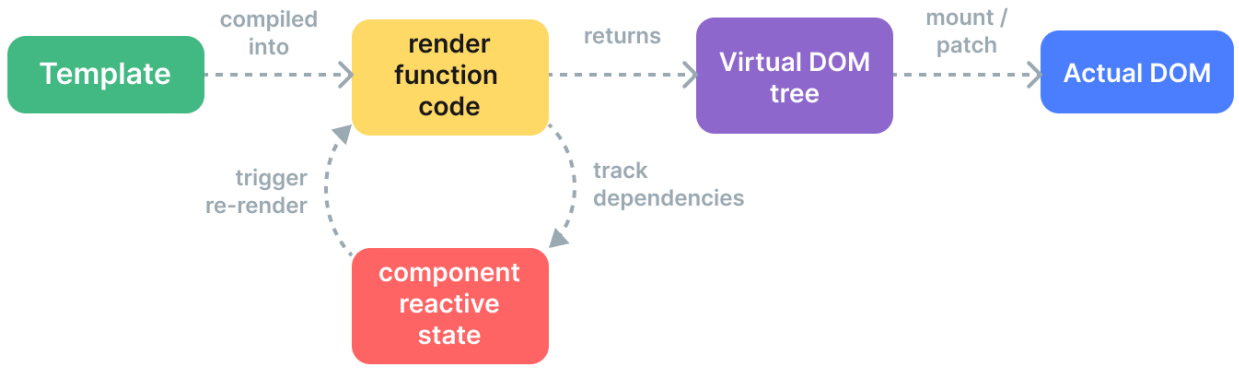
Detaljert informasjon om hver del og de forskjellige teknologier og protokoller, finnes i avsnittene under og den vedlagte systemdokumentasjonen.

4.1.1 Frontend

Løsningen som ble valgt av gruppen er en frontend-fokusert løsning som bygges i Vue.js og TypeScript, med kommunikasjon med backend for alle handlinger/funksjoner. Denne kommunikasjonen er essensiell for applikasjonen, og består av alle handlinger en ønsker å gjøre, fra å bytte kamera til å ta neste handling i en planlagt serie. For å oppnå best mulig resultat har gruppen benyttet seg av flere teknologier som komplementerer hverandre, og gir en sømløs og responsiv brukeropplevelse.

Gruppen valgte å kode i TypeScript, som er et fremadstormende (StackOverflow, 2022) programmeringsspråk som fungerer som et overlegg på JavaScript. Dette introduserer statiske typer, som medfører bedre verktøystøtte og feildeteksjon, i tillegg til at det forenkler samarbeid mellom utviklere ved å tilrettelegge for mer robust kode i et ellers dynamisk programmeringsspråk.

Ulempen med valget var den nødvendig bratte læringskurven, da dette var et språk ingen i gruppen hadde erfaring med. Det ble derfor lagt til rette for mye tidsbruk på kompetanseøkning tidlig i prosjektet, men gruppen erfarte at dette ble en trend for hele prosjektet, ettersom det stadig ble oppdaget nye problemer med nye løsninger. Valget gruppen endte med ble gjort på bakgrunn av et ønske om kompetanse innenfor et svært relevant språk, som vil være gunstig for fremtidig arbeidsliv, til tross for den ekstra tidsbruken gjennom prosjektet.



Figur 4.2 Overordnet visualisering av Vue og dens "Virtual DOM" (Vue.js, n.d.)

HTML, eller HyperText Markup Language, er et språk som tillater definisjon og plassering av elementer på en nettside. Disse elementene kan være alt fra bilder og lenker til tekst og knapper.

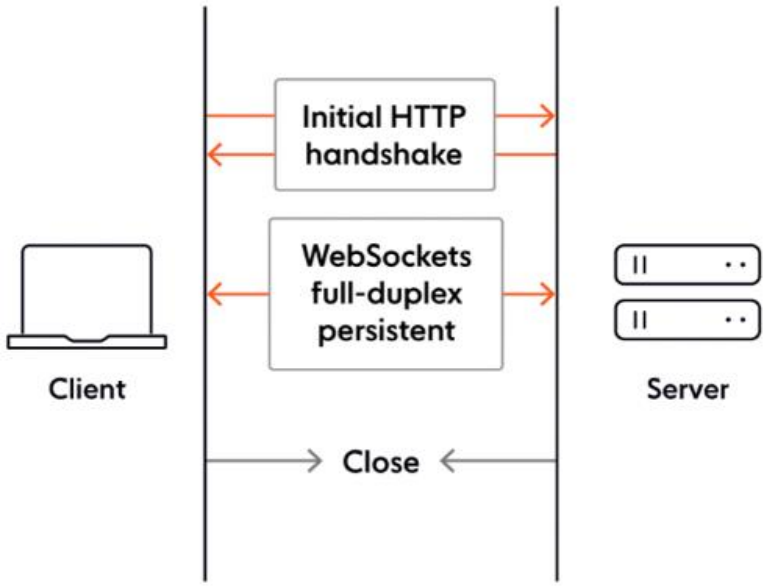
DOM, eller Document Object Model, er kort forklart en trestruktur som representerer HTML koden som lastes, når en nettside åpnes i en nettleser. DOM tillater dermed manipulasjon av elementer i denne HTML koden, via språk som JavaScript eller TypeScript.

Vue.js benytter seg av det som omtales som «Virtual DOM» når det skal tegne opp en nettside. Enkelt forklart er dette at Vue holder en «skyggekopi» som representasjon av hvordan nettsiden (Document Object Model) ser ut for brukeren, og denne synkroniseres og sammenlignes med den faktiske nettsiden. Vue kjører kontinuerlig denne sammenligningsprosessen der denne «skyggen» sammenlignes med den faktiske DOM. Ved endringer på nettsiden finner Vue ut hva som er endret via denne sammenligningsprosessen, og oppdaterer deretter *kun* det som har endret seg. Dette konseptet, illustrert i Figur 4.2, gjør at både brukeropplevelsen og utviklerens opplevelse blir smidig (Vue.js, n.d.).

4.1.2 Backend og API

Backend er en del av enhver web-applikasjon, selv der det meste av funksjonalitet skjer på frontend. For kontakt med oppdragsgivers backend-løsning som dette prosjektet skal benytte seg, ble det opprettet kontakt med deres systemer via SignalR (Microsoft, 2023), som igjen benytter seg av WebSocket (Mozilla, 2023).

WebSocket er en kommunikasjonsprotokoll som gir full duplekskommunikasjon over enkel TCP-forbindelse, som betyr at en kan både sende og motta beskjeder over internett, uten å måtte opprette en ny forbindelse for hver gang. Denne protokollen passer svært godt til vår applikasjon, ettersom den etablerte forbindelsen eliminerer behovet for å sende gjentatte forespørsler og svar for hver utveksling. Dette gjør protokollen effektiv og lite ressurskrevende, og bidrar til å minimere forsinkelse og maksimere ytelse.



Figur 4.3 Høynivå illustrasjon av en WebSocket-tilkobling (Barnes, 2022)

I kombinasjon med WebSocket ble det benyttet et bibliotek som heter SignalR, som forenkler prosessen med å opprette og implementere kommunikasjon mellom to eller flere enheter.

Biblioteket har en rekke verdifulle punkter, blant annet en automatisk fallback, som gjør at hvis WebSocket ikke er tilgjengelig, så faller SignalR tilbake til andre teknologier, slik at funksjonene fungerer på tvers av nettlesere og enheter. Denne sikkerheten, kombinert med at den håndterer automatisk opprettelse og styring av WebSocket-forbindelser mellom klient og server, ble avgjørende i valget av teknologien.

Composition API brukes for å organisere og gjenbruke logikk i Vue.js-applikasjoner ved å dele opp koden i mindre, uavhengige funksjoner. Dette gir en lettere oversikt og kontroll over appens struktur, men må benyttes med måte for å ikke komplisere det mer enn nødvendig. Script setup er en funksjon som forenkler bruken av Composition API i Single File Components (VueJs, 2023), og betyr i praksis at man kan gruppere logikken knyttet til en bestemt funksjon, noe som fører til komponenter som er mer lesbare, og enklere å vedlikeholde.

4.1.2.1 Tilkobling til SignalR-hub

Som beskrevet i avsnitt 4.1.2 skjer kontakten mellom frontend og backend via SignalR. Oppdragsgiver har bygget en SignalR-hub som står for kommunikasjonen mellom klient og server, og det er denne «hubben» sin funksjonalitet denne applikasjonen benytter seg av.

Web-applikasjonen oppretter en tilkobling til denne ved å kjøre følgende kode:

```
const connection = new signalR.HubConnectionBuilder()
    .withUrl("http://localhost:[redacted]")
    .build();

  async function start() {
    try {
      await connection.start();
      console.log("SignalR Connected.");
    } catch (err) {
      console.log(err);
      setTimeout(start, 5000);
    }
  }

  connection.onclose(async () => {
    await start();
  });

  // Start the connection.
  start();
```

Figur 4.4: kode som oppretter en tilkobling til SignalR-hubben.

Figur 4.4 viser koden som oppretter og starter tilkoblingen, og er hentet fra den offisielle dokumentasjonen fra Microsoft (Microsoft, 2023), men modifisert slik at URLen peker til Mosart-hubben bygget av oppdragsgiver.

Ved oppstart av applikasjonen kjøres denne kodesnutten slik at tilkoblingen etableres tidlig, som da gjør at annen funksjonalitet som benytter seg av dataen som sendes over denne tilkobling også kan kjøres.

En mer detaljert beskrivelse av SignalR og hva den betyr for applikasjonen er skrevet i vedlegg 4 Systemdokumentasjon, kapittel 6.

4.1.3 Applikasjonslogikk

4.1.3.1 Single-Page Application

Applikasjonen er bygget som en Single-Page Application (SPA), der det aller meste av det brukeren opplever og ser skjer i selve nettleseren. Det er en dynamisk nettside der den samme visningen oppdateres automatisk. Måten Vue fungerer ved at den «speiler» dokumentet som utgjør en nettside og oppdaterer den faktiske nettsiden ved enhver endring av dokumentet, er det som muliggjør en slik SPA. Dette er en teknologi som gjør nettsiden lettkjørt og interaktiv, og det er ikke nødvendig med videresending til andre URLer, noe som vil si at URLen i adressefeltet ikke vil endre seg etter hvert som det skjer endringer i applikasjonen mens den er i bruk. For en applikasjon som bygges i dette prosjektet, er dette en god løsning ettersom hele kontrollflaten, her definert som en tidslinje, er det som er i fokus uavhengig av hvor i applikasjonen man befinner seg.

Frontend-delen, det vil si det brukeren ser og bruker, gjerne kalt for presentasjonslaget, er koblet til et baksystem. Dette baksystemet (backend) kan utføre handlinger via meldinger/funksjonskall som sendes fra web-applikasjonen, og er en kritisk del av applikasjonen. I dette tilfellet er WebSocket benyttet som kommunikasjonsprotokoll, dette fordi kommunikasjonen og funksjonaliteten som applikasjonen skal tilby krever en to-veis kobling for å kunne både sende kommandoer og motta informasjon.

Denne backend-løsningen fungerer som et API for forskjellige applikasjoner som ønsker å utføre noe som ligger «bak» backend. APIet tilbyr endepunkter som kan kalles på for å utføre handlinger, hente ut informasjon og mye annet.

En kjernefunksjon som har blitt en viktig del av vår applikasjon, er selve «pulsen» til sendingen. Denne pulsen er et endepunkt som hvert sekund sender informasjon om statusen i Viz Mosart. Denne informasjonen er grunnsteinen i mye av det som skjer i applikasjonen og det som brukeren selv ser og opplever, deriblant tidslinjen som viser innslag, tiden og progresjonen på hvert enkelt del-innslag.

For en grundigere beskrivelse av de forskjellige endepunktene og hva de tilbyr, henvises leseren til vedlegg 4 Systemdokumentasjon, kapittel 6.

4.1.3.2 Pinia Store

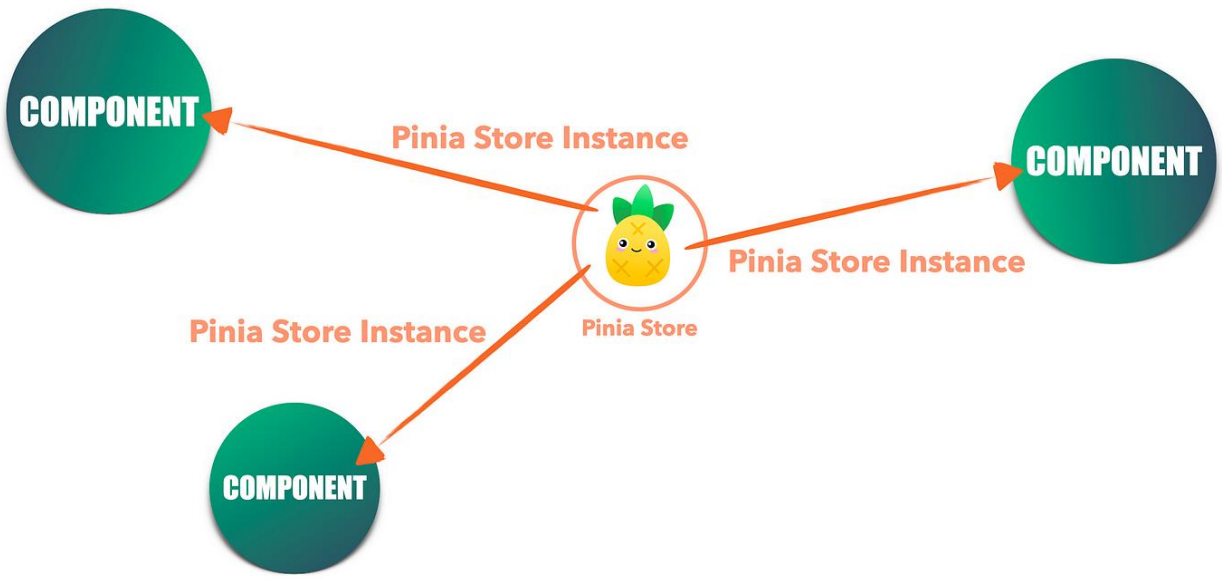
Pinia (Morote, 2019) brukes i applikasjonen som en sentral lagringsplass for å holde og dele data og applikasjonstilstand mellom Vue-komponentene. På denne måten kunne vi legge all funksjonalitet i én Pinia-fil, som enkelt kunne kobles til nødvendige komponenter. Her ble det vurdert å benytte seg av omfattende funksjoner som «prop» og «emit», men det ble skrinlagt grunnet applikasjonens smale scope og vår vurdering av den solide funksjonaliteten med nåværende nivå. Rent praktisk betyr dette oppsettet at alle nye komponenter kan implementeres med nødvendig funksjonalitet gjennom et metodekall som for eksempel «*Store.Funksjonsnavn()*», som vist i Figur 4.5.

```
import { useStore } from "@/stores/Store";
const Store = useStore();
```

```
@click="$event => Store.nextButtonToggle()"
```

Figur 4.5: import av en "Store" og metodekall på en metode i den importerte "Store"-en, hentet fra web-applikasjonens kildekode.

Årsaken bak valget av Pinia er todelt, og den mest nærliggende er hvordan programmet forenkler og løser en rekke utfordringer knyttet til tilstand og lagring av midlertidige tilstander. Ved bruken av en slik tilgjengelig lagringsplass er det også betydelig enklere å skalere opp applikasjonen for et bredere kundesegment, noe oppdragsgiver vil verdsette etter prosjektet er gjennomført.



Figur 4.6: Pinia konseptskisse (Medium.com, n.d)

4.1.4 Skalering

Et viktig fokus for både gruppen og arbeidsgiver er at applikasjonen som utvikles skal kunne utvides med mer funksjonalitet, også i fremtiden. For å sikre dette må det tas gode og gjennomtenkte valg som sikrer en levetid, og det gjerne uavhengig av hvem som utvikler koden. I det ligger det at koden skal dokumenteres godt, skrives fornuftig med optimale løsninger og smart gjennomføring, men også en forståelig logikk som kan bygges videre på. Begrepet teknisk gjeld (Kruchten, et al., 2012) er mye brukt i bransjen, og noe gruppen har hatt i tankene underveis. Dårlige valg tidlig kan sette en hindring for fremtidig skalering, spesielt hvis man må endre grunnleggende forretningslogikk på et senere tidspunkt.

4.1.5 Datahåndtering og brukerautentisering

Prosjektets struktur og funksjon tilsier at en i utgangspunktet ikke trenger en egen database for sendeplaner, men at disse vil bli hentet eksternt ved behov. Under utviklingsprosessen har disse sendeplanene blitt lastet inn i Mosart, hvor applikasjonen deretter hentet dem. Det er heller ikke nødvendig med en slik sendeplan for å bruke tjenesten, og det er i første iterasjon ikke behov for noe lagret plan i applikasjonen. Det er verdt å påpeke at Mosart ble kjørt lokalt på utviklernes egne maskiner i løpet av utviklingsfasen, men i en reell brukssituasjon for kunder ville sannsynligvis Mosart kjøre på en ekstern server eller skytjeneste.

Når det gjelder brukerautentisering, ville man i en typisk applikasjon som dette forvente at det var en brukerdatabase for innlogging og tilkobling til aktuelle sendeplaner. Dette ble ikke inkludert i gruppens prosjekt, ettersom det ikke var en kritisk del av applikasjonens funksjonalitet. Dette er et aspekt som kan implementeres i fremtidige versjoner av applikasjonen, avhengig av variasjon i krav og brukerbehov.

4.1.6 Sikkerhet

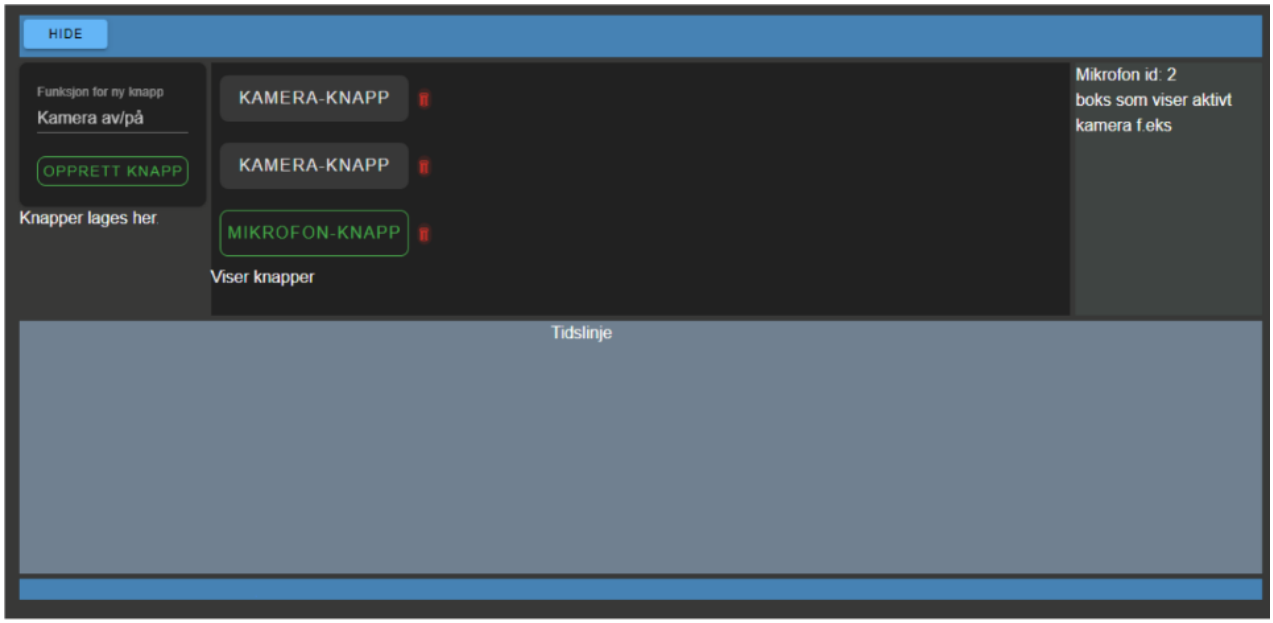
Sikkerhet er en naturlig del av enhver applikasjon som bygges, spesielt hivs applikasjonen skal brukes via internett. Det er likevel situasjoner der sikkerheten ikke settes i fokus, eller at sikkerhetsfokuset flyttes til et seinere tidspunkt i utviklingsløpet.

I løpet av utviklingen av denne applikasjonen har sikkerhet hatt lav prioritet, primært fordi produktet ikke er tilkoblet internett på dette stadiet. Sikkerheten i systemet vil heller ikke være

sterkere enn sitt svakeste ledd. Det betyr at sikkerheten i dag blir satt til eksisterende nivå, som følge av at Viz Mosart ikke har mer sikkerhet enn nettverket og serveren det kjører på.

Selv om applikasjonen i sin nåværende tilstand ikke er koblet til internett og dermed ikke er eksponert for eksterne trusler, er det viktig å tenke på hvordan situasjonen kan endre seg i fremtiden. I løpet av senere faser av prosjektet kan det bli nødvendig å legge til flere funksjoner og tilkoblingsmuligheter som innebærer samspill med eksterne systemer og brukere. Dette vil medføre nye sikkerhetsrisikoer og et økende behov for solide beskyttelsestiltak.

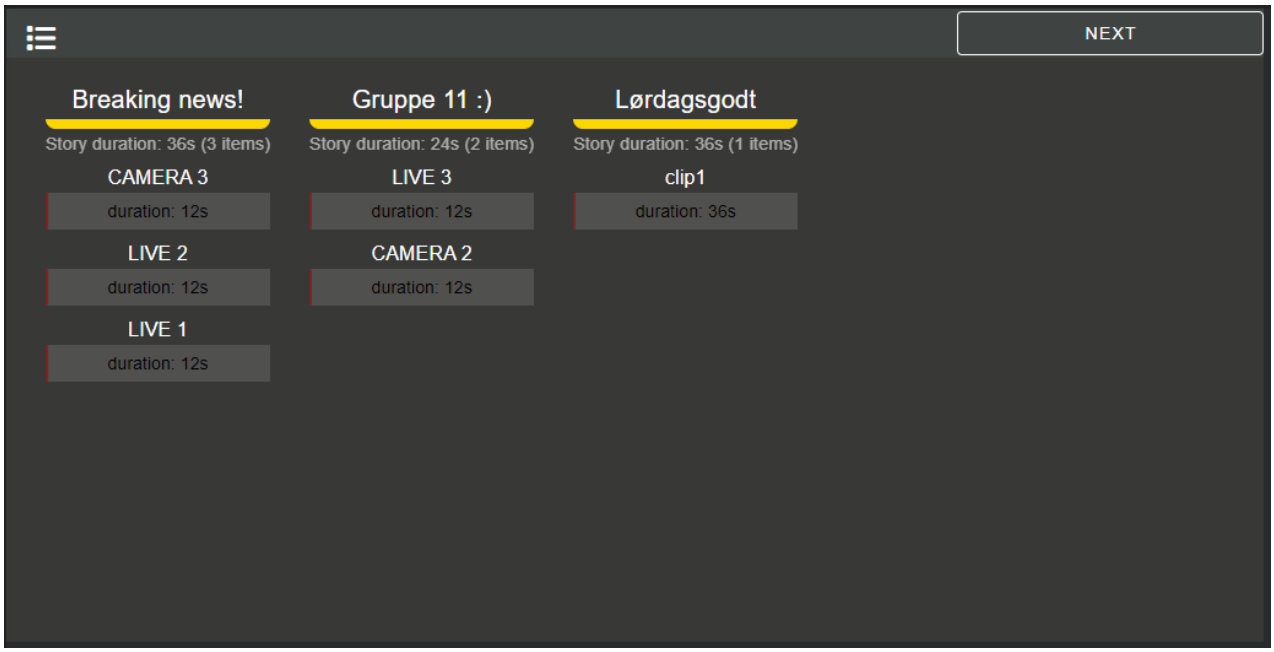
4.2 Brukergrensesnitt



Figur 4.7 en tidlig utgave av web-applikasjonen

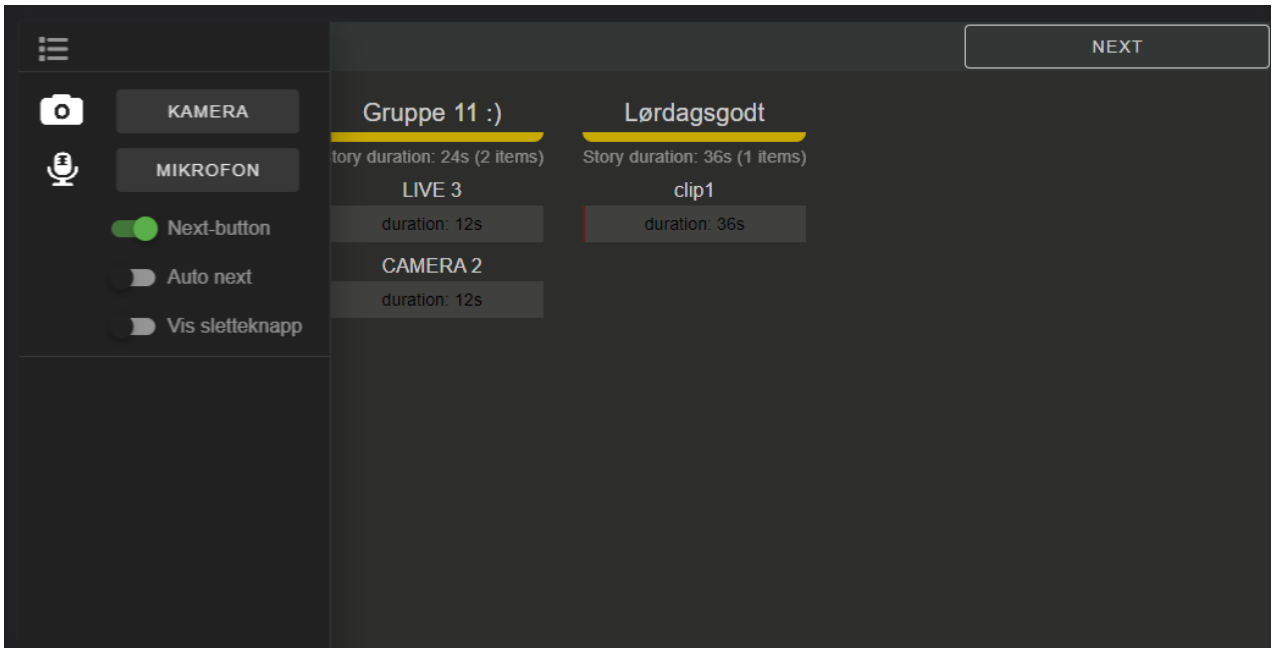
Brukergrensesnittet er forsøkt bygget i henhold til de kravene oppdragsgiver har og de gruppen selv har utarbeidet i det vedlagte visjonsdokumentet (Vedlegg 2 Visjonsdokument, kapittel 6). Krav 2.1 sier hvordan brukergrensesnittet skal være og hva som er viktig i utformingen av denne applikasjonen.

I avsnittene under beskrives det i detalj hvordan den er bygget opp og hvilken funksjonalitet som er å finne.



Figur 4.8: hovedsiden i web-applikasjonen

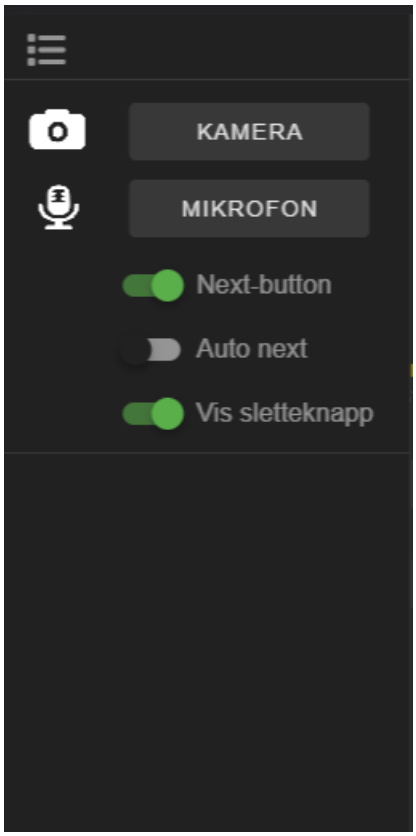
Oppe til venstre vil man finne en uttrekksmeny hvor en kan opprette knapper for de ulike datastrømmene en ønsker å bruke, samt aktivere og deaktivere flere funksjoner. Dette panelet kan skjules under bruk, slik at mer av overflaten kan brukes til tidskritiske handlinger.



Figur 4.9: uttrekksmeny med sendeplanen synlig i en dimmet farge

Ved å trykke på meny-knappen, vil uttrekksmenyen vist i Figur 4.10 skyves ut fra venstre side. Resten av applikasjonen vil da dimmes, slik at de er tydelig at det er menyen som er i fokus. Menyen kan enkelt trekkes tilbake ved å klikke hvor som helst utenfor menyen.

Dimming av området utenfor menyen er vist i Figur 4.9.

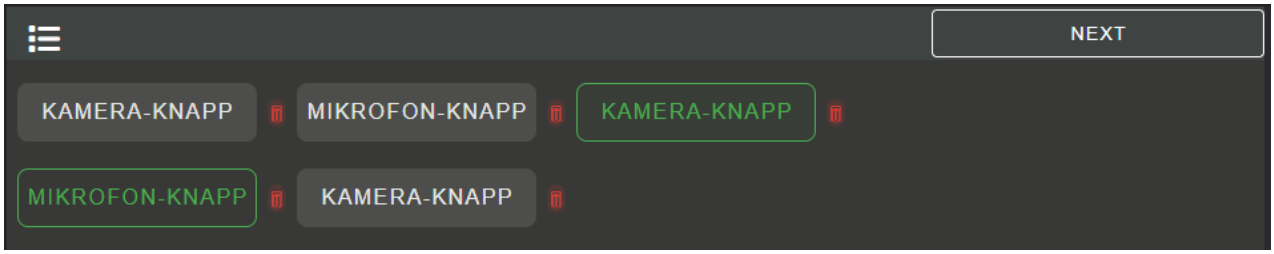


I menyen finnes det funksjonalitet for følgende:

- Legge til kamera-knapp
- Legge til mikrofon-knapp
- Vise/skjule «Next»-knappen
- Vise/skjule «Slett»-knappen
- «Auto next», planlagt funksjonalitet der denne bestemmer om det skal automatisk gjøres et hopp til neste innslag eller ikke.

Figur 4.10 uttrekksmeny, detalj. Her er både «Next»-knappen og «Slett»-knappen aktivert og vil dermed visest i applikasjonen.

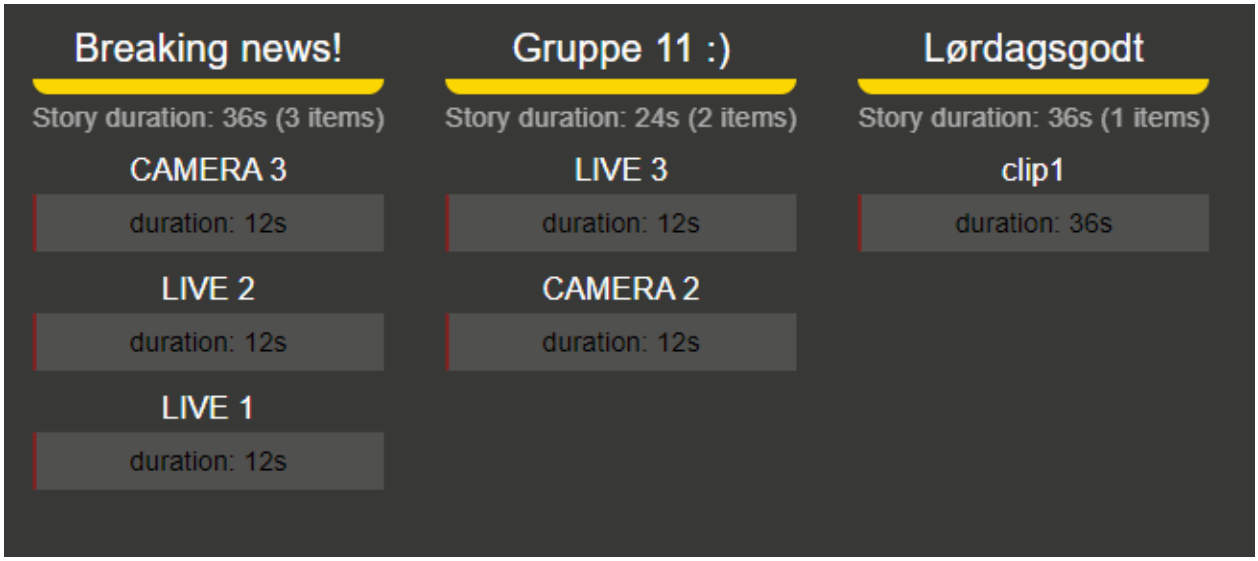
På midtpartiet i øvre del av skjermen vil det være en oversikt over de opprettede knappene (Figur 4.11), med tilhørende beskrivende tekst. Dette partiet vil være dynamisk og utvides etter hvert som det legges til flere knapper. Ved bruk av applikasjonen vil hver av disse knappene samsvare med lyd- eller videokilder som brukes i sendingen, og være koblet til selskapets backend system.



Figur 4.11 knapper, med tilhørende sletteknapp

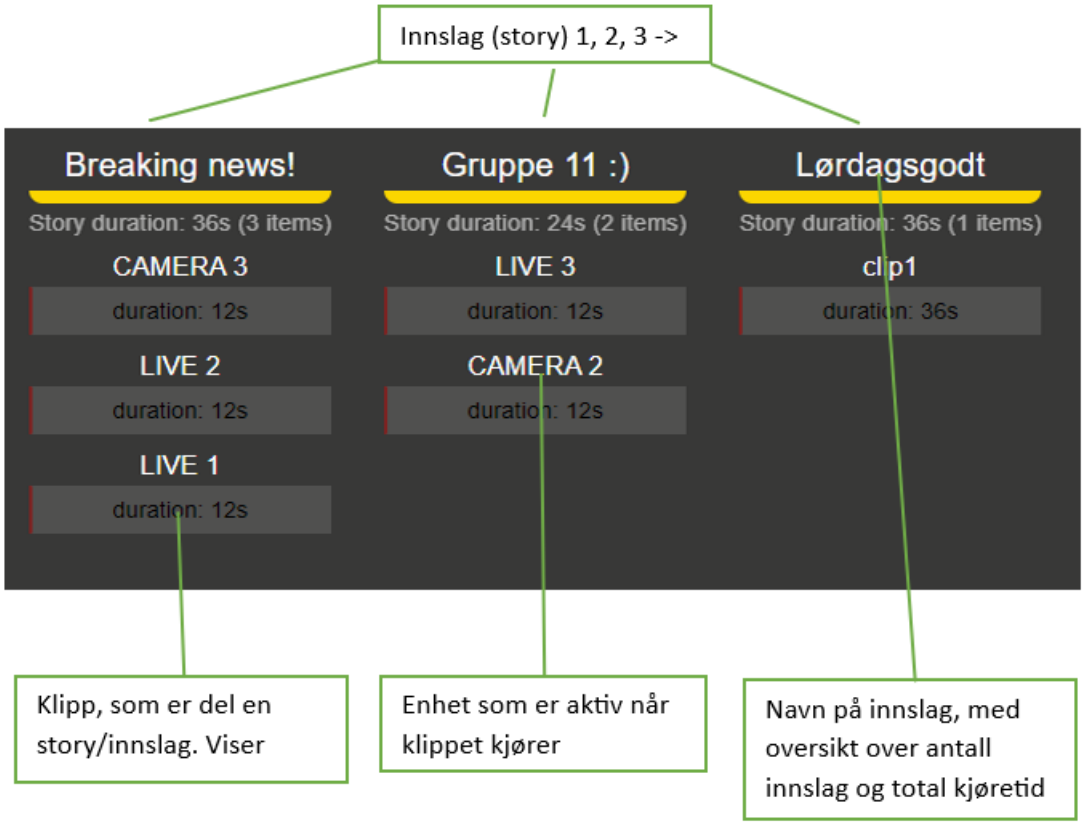
Oppe til høyre vil det være en boks som viser det aktive kameraet man sender direkte, samt lydkilden for den aktuelle sendingen. Denne funksjonen er laget for at brukeren, på enkelt vis, alltid skal ha kontroll på hvilke lyd- og bildekilder som sendes. I tillegg til dette vil brukeren også enkelt kunne se tiden på nåværende innslag, og om det har gått over sin planlagte tid. Dersom det har gått over tiden, vil teksten endre farge til rødt.

Hele den nedre delen av skjermen, vist i Figur 4.12, vil vise en tidslinje for den planlagte sendingen, som en kan hente fra det tilknyttede baksystemet.



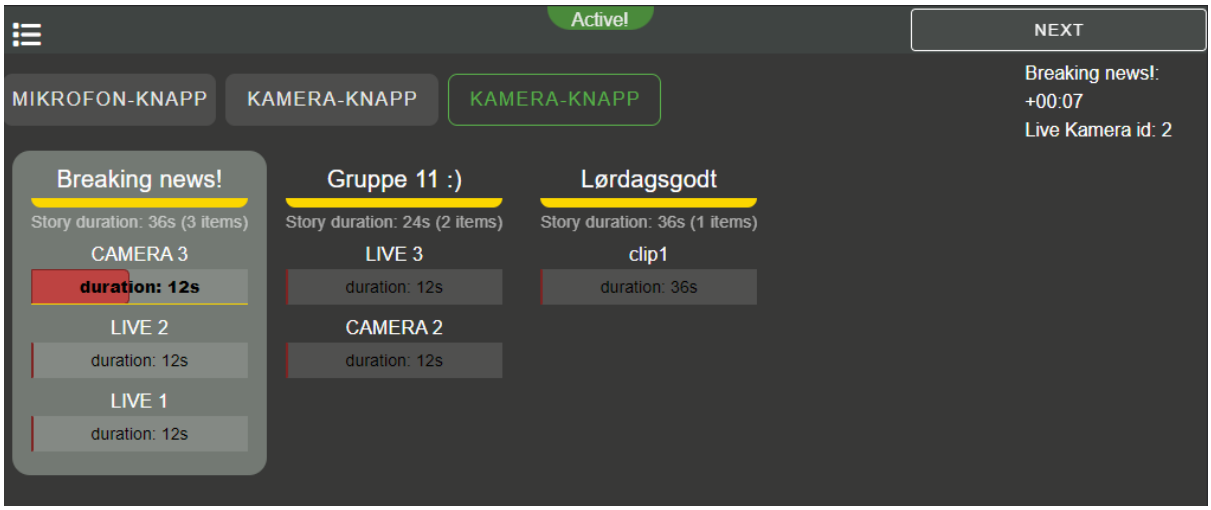
Figur 4.12: tidslinje med innlastet "Rundown" (sendeplan).

Denne tidslinjen (Figur 4.13) viser en oversikt over en pågående TV-sending, der sendingen er delt i inn i flere historier og flere klipp. Ved bruk av «Next»-knappen, plassert oppe i høyre hjørne, vil man kunne gjøre et hopp til neste historie i sendingen. Dette vil umiddelbart utføres av den tilkoblede backend-løsningen. Som bruker av applikasjonen vil man se at dette utføres ved at tidslinjen går videre til neste klipp og starter på dette.



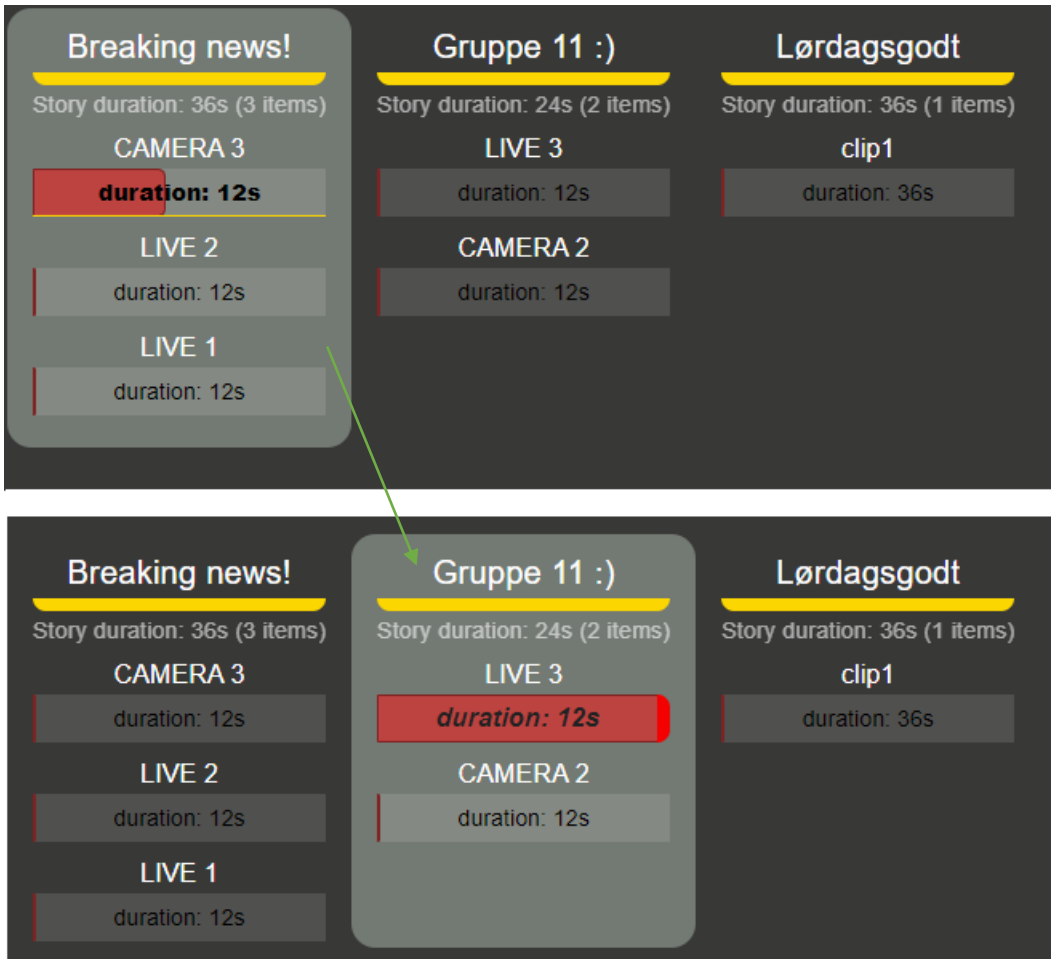
Figur 4.13 informasjonskart for tidslinjen med sendeplan

Underveis i en sending er det kritisk for brukeren at informasjonen om hva som skjer kommer tydelig frem. Her er aktivt innslag uthevet ved å fargelegge bakgrunnen. I Figur 4.14 kjører innslaget med navn «*Breaking News!*» som består av 3 forskjellige klipp, som hver varer i 12 sekunder. I nevnte figur er det første klippet i innslaget i gang, dette kan ses av den røde framdriftsindikatoren. Denne leverer data i sanntid og vil hele tiden vise hvor i klippet en befinner seg. Klippet benytter seg av «*CAMERA 3*», som vist like over denne.



Figur 4.14: tidslinje med aktivt innslag og synlige knapper.

I Figur 4.15, vist under, kan en se endringen som skjer i tidslinjen når man har trykket på «Next-knappen» for å få sendingen til å gå videre til neste klipp.



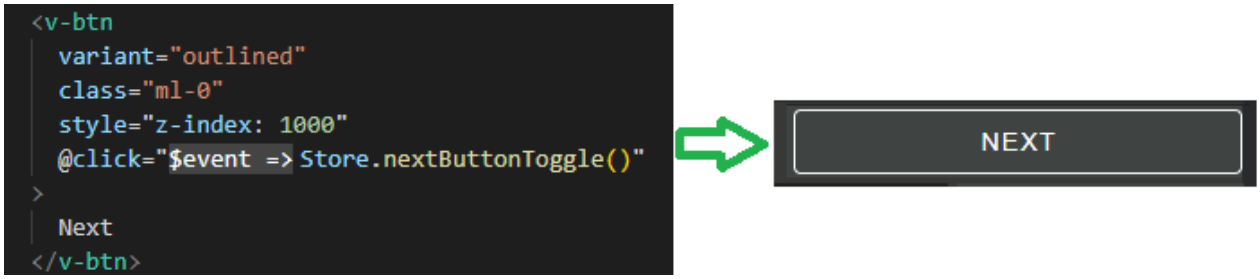
Figur 4.15: Aktive innslag. I øvre del er «Breaking news!» aktivt innslag, mens i nedre del er «Gruppe 11 😊» det aktive innslaget.

4.2.1 Komponentbiblioteker

En fordel med å utvikle i Vue.js er de mange komponentbibliotekene en kan hente komponenter fra, slik at man sparer en del grunnleggende arbeid. Det ble undersøkt og vurdert flere muligheter, men gruppen valgte å bruke Vuetify (Vuetify, 2023) på bakgrunn av deres omfattende bibliotek, i tillegg til at den følger de anerkjente prinsippene til Material Design (Google, 2023). Kjente alternativer som kunne gjort en god jobb er blant annet Tailwind CSS (Tailwind CSS, 2023) og Bootstrap (Bootstrap CSS, 2023), men ettersom arbeidsgivers egne systemer var tett integrert med Vuetify, ble dette det naturlige valget for gruppen også.

I den endelige løsningen er det flere komponenter fra Vuetify som er innhentet og tilpasset interne behov, blant annet det grafiske rundt tidslinjen med sendeplanene og flere av knappene, i form av blant annet «*v-btn*» og «*v-progress-linear*». I Figur 4.16 kan en se bilde av de ulike komponentene, samt et skjermbilde av kildekoden som generer en knapp og kobler funksjonalitet til denne.

Ved trykk på denne knappen trigges funksjonen *nextButtonToggle()*, som er lagret i applikasjonens Pinia-store.



Figur 4.16 bruk av Vuetify-komponent. Her vist med kode og hvordan de koden produserer vil se ut i nettleser

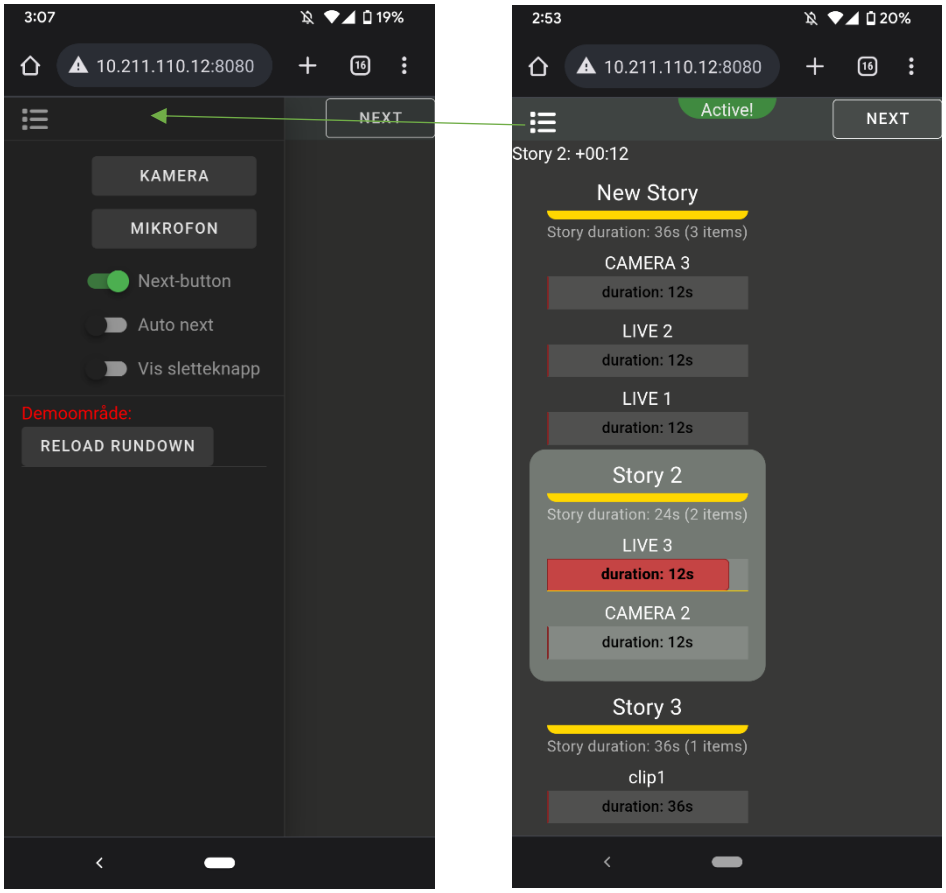
4.2.2 Bygging

Vite (ViteJS, 2023) er et verktøy for bygging og utvikling av moderne webapplikasjoner, designet av samme utvikler som står bak Vue (Evan You), og spesielt for rammeverk som Vue.js. Det tilbyr raskere bygging, gjennom teknikker som «esbuild» og «native ES modules», i tillegg til at det inneholder innebygde optimaliseringer for produksjon.

Funksjonalitet som tilbys fra Vite ble benyttet gjennom hele prosjektets utviklingsfase og har vært en viktig del i det hverdagslige arbeid, og da spesielt siden det forenkler og korter ned byggetiden til applikasjonen betraktelig. Med et slik verktøy er tiden det tar fra man lagrer koden til man kan se på resultatet i nettleser, betydelig kortere enn ved bruk av andre byggeverktøy, deriblant WebPack og Vue Command Line Interface (CLI) (Kalmanovych, 2022).

4.3 Design

I designfasen av prosjektet hadde gruppen stor grad av frihet når det gjaldt å velge den estetiske og funksjonelle retningen for applikasjonen. Det var imidlertid naturlig å søke inspirasjon fra den eksisterende skrivebordsversjonen av Viz Mosart, som inneholder den grunnleggende funksjonaliteten som gruppens løsning er tenkt, i tillegg til mye mer. Det ble også vurdert som en fordel å følge den visuelle profilen og funksjonaliteten til Viz Mosart, siden det allerede er en veletablert og gjennomprøvd løsning som er kjent for målgruppen. Ved å bevare kjente elementer og logikk fra den eksisterende løsningen, kan gruppen bygge videre på en solid plattform og dra nytte av den kollektive erfaringen og kunnskapen oppdragsgiver allerede har samlet gjennom årene med bruk og utvikling.



Figur 4.18 Applikasjon med meny på mobiltelefon *Figur 4.17 tidslinje på mobil*

I utviklingen av applikasjonen har et designprinsipp der funksjonalitet prioriteres høyt, vært sentralt for å forme produktets sluttresultat. Det er viktig å fokusere på kjernefunksjonaliteten til en applikasjon, samtidig som forstyrrende elementer minimeres og holdes på avstand (Nielsen, 2012). Med dette som utgangspunkt kom gruppen frem til at menyen skulle integreres som en uttrekkbar del, som vises i figur 4.18, slik at en enkelt kan frigjøre plass på skjermen når menyen ikke er i bruk. Denne tilnærmingen er særlig relevant for enheter med begrenset skjermstørrelse,

ettersom det er avgjørende å maksimere tilgjengelig skjermareal for den primære funksjonaliteten applikasjonen tilbyr (Wroblewski, 2011).

Et av hovedmålene for prosjektet har vært å sørge for at applikasjonen skal være intuitiv og brukervennlig på alle enheter, ettersom sluttproduktet skal være en enkel og tilgjengelig løsning. Derfor har det vært et stort fokus på visuelt synlige, grunnleggende funksjoner, og et responsivt design som utnytter størrelsen på skjermen.

Prinsippet «mobile first» går ut på å først fokusere på utviklingen av en grunnleggende løsning for liten skjerm, og deretter skalere opp til større formater og flere muligheter. Prinsippet har blitt stadig viktigere i løpet av de siste årene, spesielt når det kommer til utvikling av webapplikasjoner (Marcotte, 2011). Dette skyldes en voksende andel av internettbrukere som hovedsakelig benytter mobiltelefoner for å surfe på nettet, noe som krever at applikasjoner fungerer like godt på små skjermer som på større. Denne trenden er et resultat av endret brukeradferd, hvor et økende antall mennesker globalt, til tross for store regionale forskjeller, foretrekker å bruke smarttelefon og nettbrett for nettlesing og kommunikasjon (Statista, 2023).

4.4 Grafisk grensesnitt

4.4.1 Initielle skisser

Som beskrevet i avsnitt 3.3, ble verktøyet Figma benyttet for å skissere opp mulige løsninger. Gruppen tegnet tidlig opp noen utkast som ble benyttet som et grunnlag for hvordan det grafiske uttrykket til applikasjonen skulle se ut. Ettersom dette var tidlig, ble det gjort store endringer fra de første utkastene, men grunntanken om hvor de viktige elementene ble plassert var likevel med hele veien gjennom utviklingen.

Figur 4.19 skjermbilde av 2 tidlige skisser, tegnet i Figma

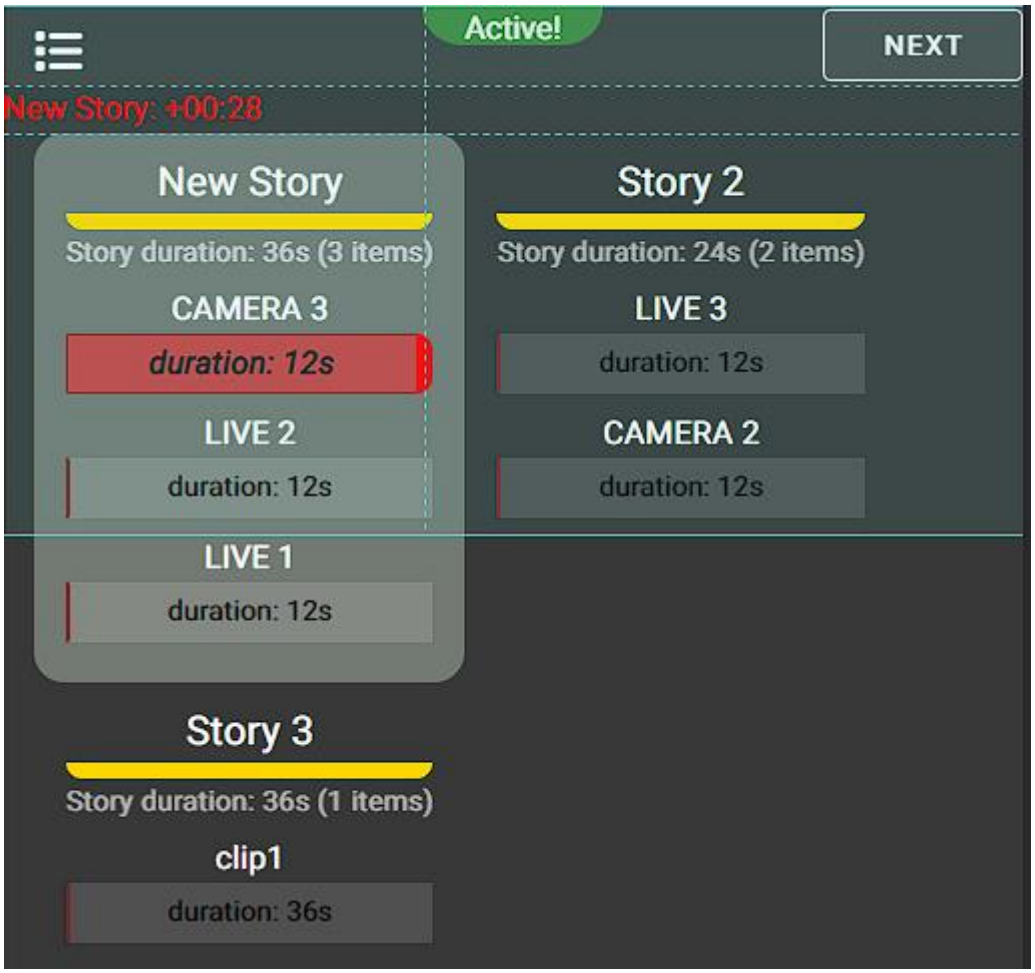
4.4.2 Oppbygging av det grafiske grensesnitt

Det ble tidlig satt et mål om høy «responsiveness», noe som betyr at applikasjonen tilpasser seg godt til ulike skjermstørrelser og enheter. Dette løste gruppen ved å definere bestemte soner i layouten, og visse regler for størrelse i korrelasjon til brukerens skjermstørrelse. For å definere de ulike sonene en vil ha i applikasjonen er det brukt CSS (Cascading Style Sheets) sin «grids»-funksjon (Mozilla, 2023), som gjør at en kan dele inn skjermen i soner med egne stiler og funksjoner. Media queries (Mozilla, 2023) er også en kjernefunksjon i styringen av web-applikasjonens utseende på de forskjellige skjermstørrelse man kan støte på. Et media query tilbyr funksjonalitet som gjør at man kan definere forskjellige grenseverdier for skjermbredde, der hver kategori har definert opp sine regler for hvordan siden skal tegnes opp av nettleseren.

```
@media (max-width: 600px) {
  #app {
    display: grid;
    grid-template-areas:
      "header header"
      "activeComponents activeComponents"
      "controls controls"
      "timeline timeline"
      "footer footer";
    grid-template-rows: 40px auto auto 200px auto;
    grid-template-columns: 211px auto;
    margin: 0;
  }
  .flexContainer {
    …..
  }
  .next {
    …..
  }
  .next2 {
    …..
  }
  .pageController {
    …..
  }
  .pageController2 {
    …..
  }
  .navBarIcon {
    …..
  }
}
```

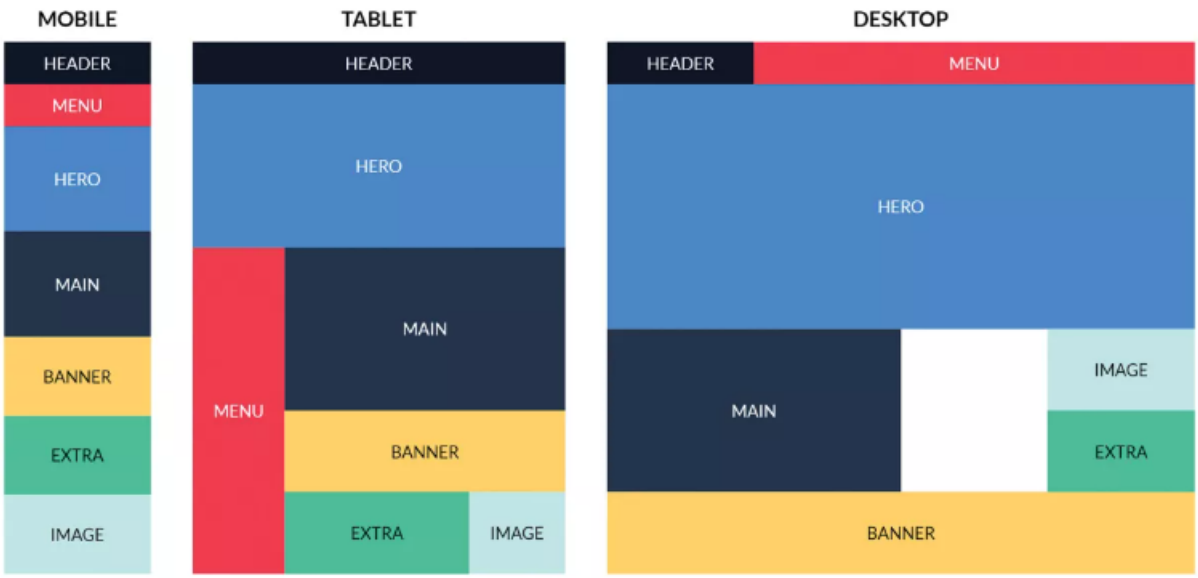
Figur 4.20: CSS media query som setter regler for skjermstørrelser med liten bredde

På bildet over kan en se en kodesnutt fra den endelige løsningen, med en maksimal bredde begrenset, sonestruktur valgt, og de fem navngitte sonene nederst. Regelsettet som er innenfor samme klamme, vil gjelde for de forskjellige komponenter og delen av applikasjonen som treffer disse vilkårene.



Figur 4.21 grid templates vist via utviklerverktøyet til Google Chrome

Figur 4.21 viser det rutenettet som er definert i media queriet vist i Figur 4.20. Det viser følgende områder markert i blått og stiplende linjer. Området «*header*» øverst, «*activecomponents*» i midten og «*timeline*» nederst. Dette forteller nettleseren hvordan den plasserer de forskjellige elementene når skjermstørrelse er under 600 piksler i bredde. Dersom skjermstørrelse går over dette, vil et annet media query med et annet sett av definisjoner og regler, styre oppsettet.



Figur 4.22 CSS grids i forskjellige skjermstørrelser (Netmag, 2021)

Figur 4.22 viser hvordan grids benyttes for å plassere elementer, eller soner, på en nettside. Nettleseren plasserer dette i henhold til reglene som er definert i CSS-filene, samtidig som den hele tiden sjekker hva reglene er for den nåværende skjermstørrelsen.

5 RESULTATER

5.1 Evalueringsmetode

Fra starten av prosjektet var planen å gjennomføre både brukertesting og enhetstesting. På bakgrunn av begrensede tidsressurser, og deretter dialog med oppdragsgiver, ble enhetstesting nedprioritert.

På grunnlag av dette ble det lagt fokus på testing i form av brukertesting. En bruker fikk teste applikasjonen i et relevant miljø der det var mulig å utføre diverse oppgaver for å måle funksjonalitet. Det ble satt opp en liste over oppgaver og funksjoner som var ønsket testet, og hvordan de skulle utføres. Slik testing foregikk både på stor og liten skjerm, slik at applikasjonens evne til å tilpasse seg forskjellige enheters skjermstørrelse ble grundig utprøvd.

Evalueringen tok for seg de viktigste punktene som gruppen tidlig definerte som avgjørende milepæler og mål. Dette ble gjort for å avgjøre om enkeltkrav ble innfridd, og minimumskravet som var forventet fra oppdragsgiver. I samråd med oppdragsgiver er ovennevnte test- og evalueringsmetoder vurdert som tilstrekkelig for å vurdere om gruppen har utviklet riktig produkt.

5.2 Evalueringsresultat

Evaluering ble utført den 4. mai 2023. Testperson var Espen Drange, fra Vizrt AS.

Generelt sett er tilbakemeldingene positive, med høye karakterer for brukervennlighet og grunnleggende funksjoner. Det opplevdes som enkelt å starte en sending og legge til knapper for mediekilder, uavhengig av skjermstørrelse eller enhet. Applikasjonen ble vurdert som pålitelig og fikk høy vurdering på responsivt design.

Det ble gitt tilbakemelding på flere områder med forbedringspotensial, blant annet knappene i applikasjonen, som kunne vært designet litt mer intuitivt. I tillegg kunne det vært enklere å knytte disse opprettede knappene til et kamera eller en mikrofon, noe som ikke er mulig på nåværende tidspunkt. Sikkerheten til applikasjonen er også vurdert som lav, ettersom dette aspektet ikke er inkludert i gruppens løsning.

Fra kravene satt tidlig i prosjektets fase er det flere som ikke er oppfylt, da disse viste seg å være for komplekse til å la seg gjennomføre innenfor prosjektets levetid. Disse er med i evalueringsskjemaet for å belyse hvilke krav gruppen valgte å prioritere vekk, for å oppnå et fungerende MVP. De ble likevel vurdert som fornuftige neste steg i applikasjonens levetid, dersom oppdragsgiver ønsker å videreføre prosjektet.

Instruksjoner: Vennligst vurder din opplevelse med applikasjonen basert på følgende krav ved å bruke skalaen 1-5, der 1 er dårligst og 5 er best. Legg gjerne igjen en skriftlig kommentar hvis du mener det er nødvendig med mer informasjon.

Sett ring rundt det som testes: Stor skjerm / Liten skjerm

Spørsmål	Vurdering (1-5)	Kommentar
Hvor enkelt kunne du legge til kontroll for potensielle bildekilder (videoklipp, kamera, bilder, grafikk)? (Krav 1.1)	4	Kanskje enda bedre med + knapp? Ellers veldig bra.
Hvor enkelt kunne du legge til kontroll for potensielle lydkilder (videoklipp med lyd, mikrofoner, lydopptak/musikk-klipp)? (Krav 1.2)	4	Kanskje enda bedre med + knapp? Ellers veldig bra.
Hvor enkelt kunne du endre på arbeidsområdet i applikasjonen? (Krav 1.5)	2	Ikke veldig enkelt, ligger i menyer, og knapper kan ikke endre rekkefølge.
Hvor enkelt kunne du hente informasjon om planlagte scener i en sendeplan? (Krav 1.6)	4	

Figur 5.1 Utdrag fra evalueringsskjema

5.2.1 Validering

Som del av valideringen har det vært statusmøter med oppdragsgiver underveis der fremgang har blitt presentert. Basert på tilbakemeldinger fra oppdragsgiver fra disse møtene, har kursen for videre arbeid blitt staket ut. Dette har også vært nyttig for å avklare om valgene som har blitt vurdert underveis er fornuftige eller ikke, og for å unngå at det utvikles i feil retning.

Gjennom utviklingsperioden ble kompleksiteten bak gruppens krav og visjoner stadig oppdaget, og som en konsekvens ble visse krav nedprioritert. Gruppens hovedmål var å levere en fungerende applikasjon og grunnleggende funksjonalitet ble derfor prioritert over en rekke mer stilige tilleggsfunksjoner. For å nå dette målet ble en omfattende testing og opprettelse av sikkerhetsprosedyrer neglisjert, noe gruppen anerkjenner må gjøres før en slik løsning kan settes i produksjon.

5.3 Prosjektresultat

I samråd med oppdragsgiver er gruppen tilfredsstilt med sluttresultatet, til tross for en ulik løsning enn først planlagt. Kompleksiteten til den tidlige visjonen ble etter hvert vurdert til for omfattende, og gruppen avgrenset deretter. Det ble derfor konkludert med at en fungerende applikasjon med de grunnleggende funksjonene var tilstrekkelig.

Denne prototypen kan legge grunnlaget for selskapets videre utvikling innenfor et voksende marked, og bidro til et nytt perspektiv på sentrale funksjoner i deres tjenester. Resultatet ble noe annerledes enn oppdragsgiver forventet, med et større fokus på frontend og den visuelle opplevelsen av tjenesten. Web-applikasjonen fungerer som en solid prototype for videreutvikling av tjenesten, og er et resultat som oppfyller oppdragsgivers opprinnelige ønske for prosjektet.

5.4 Prosjektgjennomføring

Prosjektet ble gjennomført i tidsperioden januar 2023 til mai 2023.

Akkumulerte og ukentlige timelister med statusrapporter er tilgjengelig i den vedlagte prosjekthåndboken (vedlegg 1 Prosjekthåndbok, kapittel 4). Der finner man også risikogjennomgang med eventuelle tiltaksplaner, samt de forskjellige utgavene av fremdriftsplanen gruppen har fulgt.

Gruppen har helt siden oppstarten fokusert på jevn arbeidsmengde og fremgang gjennom hele prosjektet. For å forbedre forståelse for oppgaven og hva hver enkelt måtte gjøre, holdt gruppen ukentlige statusmøter i oppstartsfasen. Det har likevel dukket opp flere utfordringer som har påvirket arbeidet. Tekniske utfordringer, bratte lærekurver og distraksjon fra andre fag har gjort at enkelte deler av prosjektet tok lenger tid en tenkt.
At tilkobling til backend ikke skjedde før i midten av mars, gjør at prosjektets fremgang gikk noe saktere enn planlagt i den kritiske midt-perioden. Dette har påvirket det endelige resultatet av prosjektet, blant annet ved at nødvendige tidsavgrensninger hindret utforsking av noe funksjonalitet som ville passet bra i applikasjonen.

En annen utfordring som har påvirket prosjektgjennomføringen er vedvarende sykdom hos gruppemedlemmer. Det har vært flere runder med sykdom for flere av gruppens medlemmer, noe som har tidvis hindret fysiske møter og midlertidig begrenset den kvantitative innsatsen. Det ble tidlig lagt til rette for fleksibel arbeidstid og sted, og gruppen har kommunisert tydelig hva som forventes av bidrag gjennom ulike perioder. Den største påvirkningen har vært å innføre digitale hjelpeverktøy innenfor planlegging, kommunikasjon, og samarbeid via internett. Gruppen mener at tiltakene har begrenset de negative konsekvensene sykdomsfravær kunne hatt, og samtidig sørget for en god moral hos medlemmene gjennom hele prosjektet.

6 DISKUSJON

Gjennom prosjektets løp har gruppen måtte foreta en rekke valg i forhold til metoder og arbeidsprosesser. I denne drøftingen vil vi utbrodere disse valgene, diskutere rundt potensielle forbedringer, og vurdere de reelle konsekvensene av valgene som ble tatt.

6.1 Konsekvenser av valg

En av de merkbare konsekvensene av å velge en teknologisk stack med høy kompleksitet og en bratt læringskurve (TypeScript og Composition API) er at det har tatt lengre tid enn forventet å implementere funksjonalitet. Dette kan delvis tilskrives den tidkrevende prosessen med å utvikle kompetansen innenfor valgte emner, før vi kunne begynne å kode effektivt. Som et resultat av dette har prosjektets fulle potensial vært noe begrenset, i forhold til om vi hadde benyttet oss av programmeringsspråk som gruppen allerede mestret.

I løpet av utviklingen av prosjektet ble testingen av applikasjonen mindre omfattende enn vi først hadde planlagt. Flere faktorer bidro til dette, blant annet implementeringen av applikasjonen som en «Single Page Application», og bruken av «hot reloading» i utviklingsprosessen. Dette førte til hurtig feedback i utviklingsprosessen, og medvirket i at utviklerne oppdaget problemer i koden raskt. Videre hadde prosjektets tidsramme innvirkning på testingen, ettersom gruppen var nødt til å prioritere implementering av funksjonalitet og tilhørende nødvendig utvikling av egen kompetanse. Dette resulterte i begrenset tid til å utforme, utvikle og gjennomføre omfattende tester, noe som kan påvirke kvaliteten og ytelsen til applikasjonen.

Det er viktig å erkjenne at omfattende testing er en avgjørende del av en utviklingsprosess til applikasjoner, og at utilstrekkelig tid og ressurser til testing kan resultere i et produkt av lavere kvalitet. For kommende prosjekter vil gruppens medlemmer ta med seg erfaring fra dette, og vurdere potensielle utfordringer knyttet til ny teknologi, og tilpasse ressursfordeling og testplaner deretter.

6.2 Fordeler

Et nøkkelaspekt ved vårt prosjekt har vært etableringen av et tidlig og realistisk mål. Ved å ha et tydelig bilde av ønsket funksjonalitet tidlig i prosessen, som for eksempel «neste»-knappen, samt tidlig inkludering av kamera- og mikrofonknapper, kunne vi oppnå synlige resultater som bidro til en motiverende mestringsfølelse.

Ved å benytte TypeScript og Composition API med <script setup> har vi valgt en fremtidsrettet og relevant tilnærming, som resulterer i en applikasjon som kan videreutvikles og brukes i

fremtiden. Den bratte læringskurven for utviklerne har båret frukter, da gruppen anser den begrensede, men fremtidsrettede applikasjonen som gunstigere enn en alternativ versjon med mer funksjonalitet, men som vil falme raskere.

Det grafiske designet og brukeropplevelsen anser gruppen som et av prosjektets største styrker. Tidlig ble dette et primært fokus, gitt oppgavens klarhet om å forenkle tilgjengeligheten til et fungerende backend system. Ved å utvikle applikasjonen i Vue har vi skapt en moderne og engasjerende brukeropplevelse, noe som er helt essensielt for å tiltrekke seg og beholde sluttbrukere.

Samarbeidet og arbeidsmetodene mellom gruppens medlemmer spiller også en viktig rolle i prosjektets suksess. Selv om dette ikke er direkte relatert til de valgte metodene, har åpen kommunikasjon og godt samarbeid mellom medlemmene bidratt til å løse problemer og håndtere utfordringer som har oppstått underveis.

6.3 Ulemper

Gjennom prosjektet har gruppen støtt på noen utfordringer relatert til valg av prosess, metode og teknologi. Dette har ført til at visse aspekter kunne ha vært forbedret, men har ført til viktig læring for studentene.

Først og fremst kunne vi trolig ha nådd dagens funksjonalitetsnivå raskere hvis vi hadde valgt en enklere teknologistack. Det vi endte med var å benytte Composition API med <script setup> og TypeScript, noe som kan være krevende for nybegynnere. Selv om det finnes mer hjelp og bedre ressurser for Options API og uten script setup, mente vi at det var teknologisk fornuftig å velge denne mer avanserte løsningen. Dette skyldes at gruppen hadde et ønske om å maksimere læringsutbyttet, samt posisjonere oss godt i et nærliggende fremtidig arbeidsmiljø. Gruppen er derfor nødt til å erkjenne at dette valget kan ha ført til en lengre læringskurve og dermed begrenset det potensielle omfanget av sluttproduktet.

Videre kunne vi muligens ha nådd noen av målene tidligere hvis vi hadde hatt en mer strukturert tilnærming til Scrum, som ble nevnt tidlig i prosjektfasen. Vi kunne vært jevnere og tydeligere, særlig i perioden etter midtveisrapporten, på milepæler og oppgaver, og med det sørget for en bedre organisasjon og fordeling av arbeidsoppgaver. Selv om gruppen hadde frekvente interne møter, anerkjenner vi at effektiviteten og produktiviteten kunne potensielt ha blitt forbedret ved å ha et større fokus på å følge Scrum-prinsippene.

I oppsummering viser det seg at våre valg og metoder har medført både positive og negative konsekvenser for prosjektets resultat. Det å lære av disse erfaringene og bruke dem i fremtidige prosjekter vil bidra til å forbedre både arbeidsprosessen og sluttresultatene. Eksempelvis kan vi være mer selektive i valg av teknologi, balansere kompleksitet og brukervennlighet, samt prioritere kommunikasjon med oppdragsgivere og prosjektledere. Ved å benytte mer

strukturerte utviklingsmetoder og sikre at tilstrekkelig tid er avsatt til testing og kvalitetssikring, vil vi være bedre forberedt til å levere produkter med høy kvalitet ved fremtidige prosjekter. En nøkkelfaktor for dette blir å reflektere over erfaringene vi har fått, og være periodisk selvkritiske for å sikre kontinuerlig forbedring i vårt arbeid.

6.4 Forbedringer

Sluttproduktet bærer noe preg av en visjon som ble utarbeidet på et tidspunkt der gruppens medlemmer ikke var helt klar over oppgavens omfang. Konsekvenser av dette er blant annet at fremdriftsplanen settes opp uten at man helt vet hvor lang tid hver oppgave tar, noe gruppen fikk fort erfare.

Som en forbedring her burde gruppen gjort grundigere undersøkelser før krav og visjon ble satt. Dette kunne vært gjort på flere måter, men i hovedsak ved tettere kontakt med oppdragsgiver i den tidlige fasen. I tillegg kunne gruppen lagt enda større fokus på læring av nye teknologier i oppstartsfasen, da økt kunnskap gir bedre grunnlag til å vurdere tidsbruk på kravene som ble satt opp.

En annen mulig forbedring kunne ha vært å følge Scrum tettere, for å sikre en jevn arbeidsflyt og oftere oppdaterte delmål i prosjektet. Selv om vi i dette prosjektet endte opp med en slags preferert hybrid av Scrum og iterativ utvikling, spekuleres det om gruppen kunne dratt nytte av å etablere en mer systematisk tilnærming til prosjektledelse og utvikling.

7 KONKLUSJON OG VIDERE ARBEID

7.1 Konklusjon

Hovedmålet for oppgaven var å lage en web-applikasjon som kan styre en medie-sending. Sluttproduktet oppfyller de grunnleggende kravene for oppgaven, men mangler noe funksjonalitet for gjennomføring i praksis.

Det grunnleggende rammeverket og grafiske brukergrensesnittet er implementert og utviklet i henhold til flere av de initielle kravene. Grunnet oppgavens brede omfang, har nødvendige avgrensninger gjort at noen krav ikke er implementert. I tillegg har det vist seg at backend-løsningen ikke tilbyr nødvendig funksjonalitet for å innfri noen krav som opprinnelig var definert. Det har likevel ikke hindret utviklingen av et produkt som innfrir oppdragsgivers minstekrav i dette prosjektet og dets utviklingsløp.

Følgende funksjonalitet er implementert:

- Next-knappen
- Timeline/sendingsplan
- Innhenting av sendeplan fra backend
- Mulighet til å opprette knapp for kamera
- Mulighet til å opprette knapp for mikrofon
- Sletting av knapper

På bakgrunn av denne funksjonaliteten vurderes det som at gruppens første delmål er innfridd, ettersom en sendeplan kan leses inn og gjennomføres via web-applikasjonen. Det neste delmålet omhandler utforming av eget arbeidsområde, og er et delmål sluttproduktet ikke oppfyller fullstendig.

Manglende funksjonalitet inkluderer blant annet muligheten til å justere plassering og størrelser på knappene som styrer kamera- og mikrofoner. Det ble også diskutert muligheten for en forhåndsvisning av kamerafeeden til de ulike kameraene, men det ble etter hvert konkludert med at de grunnleggende funksjonene måtte prioriteres.

Basert på evalueringene konkluderes det med at produktet ikke oppfyller alle funksjonelle mål, men gir et godt grunnlag for videre utvikling.

7.2 Videre arbeid

Videre arbeid vil kunne inkludere utvidelse av funksjonalitet og robusthet, visuelle oppdateringer, og omstrukturering av koden.

På funksjonalitetssiden er det flere punkter der applikasjonene kan utvides. Støtte for å automatisk hente ut antall kamera, mikrofoner og andre enheter som er tilgjengelig på aktuell sending, vil være noe som kan bli svært nyttig. Funksjonalitet for å styre lydnivået på en mikrofon er også noe som anbefales ved fremtidig videreutvikling.

Som nevnt i konklusjonen hadde gruppen et innledende mål om å kunne skreddersy sitt eget arbeidsområde, noe som ikke ble fullstendig oppnådd. For det visuelle er det i dag ofte ønskelig å kunne velge sitt eget fargetema, og da i det minste mellom et lyst og mørkt tema. Dette, kombinert med andre bruker-spesifikke innstillinger og alternativer, vil kunne gjøre den generelle brukeropplevelsen enda bedre i fremtiden.

Omstrukturering av koden er noe som hele tiden vil være aktuelt, spesielt når det legges til eller endres på funksjonaliteten. Ettersom dette er en applikasjon som er utviklet som et prosjekt der læring har stått i fokus, er det naturlig at nivået og kvaliteten på koden kan forbedres. Det være seg en mer fornuftig tilnærming til komponenter og abstraksjoner, eller bare måten og formuleringen en metode eller funksjon er skrevet på.

Robustheten til koden kan alltid forbedres. I en utviklingsprosess der tiden er en begrensende faktor må det tas valg som kan bety at deler av koden får mindre prioritet. Web-applikasjonen som er utviklet gjennom dette prosjektet er i utgangspunktet utviklet for å fungere i et bestemt miljø. Sikring av dette miljøet og eventuelle utfordringer som kan dukke opp dersom den skal brukes utenfor et tilpasset miljø, kan være viktige deler å forbedre i fremtiden. Som del av dette er det også sterkt anbefalt at det utvikles og skrives gode enhetstester og ende-til-ende-tester.

For fremtidige utviklingsprosjekter vil gruppens anbefaling være å tenke nøye gjennom valgene av teknologi og språk. Det bør gjøres vurdering på om umiddelbart resultat eller langsiktig resultat er mest viktig for gruppen, og hva de selv ønsker å få av kunnskap gjennom prosessen. En bratt læringskurve som ikke umiddelbart gir synlig fremdrift, kan virke demotiverende og dermed gjøre jobben betydelige tyngre enn den trenger å være.

Enda tettere kontakt med oppdragsgivers ekspertise og rådgivere er også noe gruppen vil anbefale, spesielt i starten dersom det er mye nytt å sette seg inn i.

8 REFERANSER

Barnes, E. L., 2022. *Laravel-news.* [Internett]
Available at: https://laravel-news.com/websocket-handbook
[Funnet 02 Mai 2023].

Beck, K. et al., 2001. *https://agilemanifesto.org/.* [Internett]
Available at: https://agilemanifesto.org/
[Funnet 16 Mai 2023].

Bootstrap CSS, 2023. *Bootstrap CSS.* [Internett]
Available at: https://getbootstrap.com/docs/3.4/css/
[Funnet 22 April 2023].

Discord, 2023. *Discord.* [Internett]
[Funnet 18 April 2023].

Figma, n.d.. *Figma.* [Internett]
Available at: https://www.figma.com/
[Funnet 23 Februar 2023].

Gitlab, 2014. *Gitlab.* [Internett]
Available at: www.gitlab.com
[Funnet Januar 2023].

Google Drive, u. å.. *Google Drive.* [Internett]
[Funnet 18 April 2023].

Google, 2023. *Material Design.* [Internett]
Available at: https://m3.material.io/
[Funnet 21 April 2023].

Kalmanovych, A., 2022. *BetterProgramming.* [Internett]
Available at: https://betterprogramming.pub/is-vite-really-faster-than-webpack-b414f6cc751c
[Funnet 26 April 2023].

Kruchten, P., Nord, L. R. & Ozkaya, I., 2012. *Computer.org.* [Internett]
Available at:
https://www.computer.org/csdl/magazine/so/2012/06/mso2012060018/13rRUyoyhMp
[Funnet 27 April 2023].

Macrae, C., 2018. *Vue.js: Up and Running: Building Accessible and Performant Web Apps.* s.l.:O'Reilly Media, Inc.

Marcotte, E., 2011. Responsive Web Design. I: s.l.:s.n., p. 153.

Medium.com, n.d. *Medium.com.* [Internett]
Available at: https://medium.com/geekculture/emergency-pinia-course-7a80b8ed0b04
[Funnet 26 April 2023].

Meta, 2023. *Facebook.* [Internett]
Available at:

https://www.facebook.com/business/help/1884140525218868?id=1123223941353904
[Funnet 19 Mai 2023].

Meta, 2023. *React.* [Internett]
Available at: https://reactjs.org/
[Funnet 21 Februar 2023].

Microsoft, 2023. *Microsoft dot.net.* [Internett]
Available at: https://dotnet.microsoft.com/en-us/apps/aspnet/signalr
[Funnet 25 April 2023].

Microsoft, 2023. *Microsoft Learn.* [Internett]
Available at: https://learn.microsoft.com/en-us/aspnet/core/signalr/javascript-client?view=aspnetcore-7.0&tabs=visual-studio-code
[Funnet 25 April 2023].

Microsoft, 2023. *TypeScript.* [Internett]
Available at: https://www.typescriptlang.org/why-create-typescript
[Funnet 21 Februar 2023].

Microsoft, 2023. *TypeScript.* [Internett]
Available at: https://www.typescriptlang.org/
[Funnet 21 Februar 2023].

Microsoft, 2023. *Visual Studio Code.* [Internett]
Available at: https://code.visualstudio.com/
[Funnet 21 Februar 2023].

Morote, E. S. M., 2019. *Pinia Vue.JS.* [Internett]
Available at: https://pinia.vuejs.org/
[Funnet 27 April 2023].

Mozilla, 2023. *Mdn web docs.* [Internett]
Available at: https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout
[Funnet 18 April 2023].

Mozilla, 2023. *MDN Web Docs.* [Internett]
Available at: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API
[Funnet 25 April 2023].

Mozilla, 2023. *Mdn webdocs.* [Internett]
Available at: https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries
[Funnet 22 April 2023].

Mozilla, 2023. *Mozilla WebDocs.* [Internett]
Available at: https://developer.mozilla.org/en-US/docs/Glossary/SPA
[Funnet 27 April 2023].

Nelson, B., 2018. *Getting to Know Vue.js: Learn to Build Single Page Applications in Vue from Scratch.* s.l.:Apress.

Netmag, 2021. *CreativeBloq.* [Internett]
Available at: https://www.creativebloq.com/advice/a-comprehensive-guide-to-using-css-

grid
[Funnet 24 April 2023].

Nielsen, J., 2012. *NNGroup.* [Internett]
Available at: https://www.nngroup.com/articles/usability-101-introduction-to-usability/
[Funnet 21 April 2023].

Scrum.org, 2023. *What is Scrum?.* [Internett]
Available at: https://www.scrum.org/resources/what-is-scrum
[Funnet 21 Februar 2023].

Stack Overflow, 2022. *Stack Overflow Dev Survey.* [Internett]
Available at: https://survey.stackoverflow.co/2022/#most-loved-dreaded-and-wanted-webframe-want
[Funnet 21 Februar 2023].

StackOverflow, 2022. *StackOverflow.* [Internett]
Available at: https://survey.stackoverflow.co/2022/#section-most-loved-dreaded-and-wanted-programming-scripting-and-markup-languages
[Funnet 02 Mai 2023].

Statista, 2023. *Statista.* [Internett]
Available at: https://www.statista.com/statistics/306528/share-of-mobile-internet-traffic-in-global-regions/
[Funnet 20 April 2023].

Tagline Infotech LLP, 2023. *Tagline Infotech.* [Internett]
Available at: https://taglineinfotech.com/why-vue-js-is-the-future-of-front-end-development/
[Funnet 02 Mai 2023].

Tailwind CSS, 2023. *Tailwind.* [Internett]
Available at: https://tailwindcss.com/
[Funnet 22 April 2023].

Trello, n.d. *Trello.* [Internett]
Available at: www.trello.com
[Funnet 23 Februar 2023].

ViteJS, 2023. *https://vitejs.dev/.* [Internett]
Available at: https://vitejs.dev/
[Funnet 22 April 2023].

Vizrt Group, 2022. *Vizrt.* [Internett]
Available at: https://www.vizrt.com/products/viz-mosart
[Funnet 22 Februar 2023].

Vizrt, 2023. *Vizrt.com.* [Internett]
Available at: https://www.vizrt.com/wp-content/uploads/2422-1024x576.jpg.webp
[Funnet 16 Mai 2023].

Vue.js, 2021. *Vue.js: The Progressive JavaScript Framework.* [Internett]
Available at: https://vuejs.org/
[Funnet 21 Februar 2023].

Vue.js, n.d.. [Internett]
Available at: https://vue-loader.vuejs.org/guide/hot-reload.html#state-preservation-rules
[Funnet 22 Februar 2023].

Vue.js, u.d. *Vuejs.org.* [Internett]
Available at: https://vuejs.org/guide/extras/rendering-mechanism.html#virtual-dom
[Funnet 02 Mai 2023].

VueJs, 2023. *VueJs SFC.* [Internett]
Available at: https://vuejs.org/api/sfc-spec.html
[Funnet 13 April 2023].

Vuetify, 2023. *VuetifyJS.* [Internett]
Available at: https://vuetifyjs.com/en/
[Funnet 22 April 2023].

Wroblewski, L., 2011. *Mobile First.* 1 red. s.l.:s.n.

Zaporozhets, D. & Sijbrandij, S., 2014. *Gitlab.com.* [Internett]
Available at: www.gitlab.com
[Funnet Januar 2023].

9 VEDLEGG

Vedlegg 1 Prosjekthåndbok

Vedlegg 2 Visjonsdokument

Vedlegg 3 Kravdokument

Vedlegg 4 Systemdokumentasjon

10 ORDLISTE

Ord	Forklaring
Vue.js	Et progressivt JavaScript-rammeverk
TypeScript	Programmeringsspråk som bygger på JavaScript.
HTML	HyperText Markup Language, brukes for å definere elementer plassering på nettside.
CSS	Cascading Style Sheets, brukes til å definere utseende på en HTML-fil
Vuetify	Komponentbibliotek for Vue
Vite	Byggeverktøy som lar utviklere kjøre en Vue-applikasjon, f.eks i nettleseren
Single File Component (SFC)	Fil sammensatt av flere deler: HTML, CSS og TypeScript/JavaScript. Vue benytter seg av denne typen filer.
Frontend	Den visuelle presentasjonen brukeren ser og handler med.
Backend	Baksystemet, delen som av systemet som ligger under presentasjonslaget.
GUI	Graphical User Interface – det grafiske brukergrensesnittet
API	Application Programming Interface. Tilgjengeliggjør funksjonalitet mellom komponenter i et system, f.eks mellom backend og frontend.
XML	Filformat av typen Extensible Markup Language. En rundown (sendeplan) i Mosart er levert på dette formatet.
Viz Mosart	Vizrt sin automasjons-backend. Utfører handlinger sendt fra kontrollenheter.
Rundown	En sendeplan som inneholder stories, eller klipp, som utgjør f.eks en nyhetssending.
Story (innslag)	Et innslag i en sending. Flere stories kan utgjøre en rundown.
WebSocket	WebSocket er en kommunikasjonsprotokoll som gir full to-veis kommunikasjon over en enkelt TCP-tilkobling. Protokollen er ideell for tjenester som trenger levering av sanntidsdata.
SignalR	Programvarebibliotek som tilbyr sanntidsfunksjonalitet for web-applikasjoner ved å la serveren sende innhold til klienten, når det skjer. SignalR benytter seg av WebSocket, gitt at dette er tilgjengelig.
VSCode	Microsoft Visual Studio Code. Programvare som benyttes utvikling.
Gitlab	Kodeversjoneringsverktøy, alternativ til mer kjente GitHub
Hot-reloading	Teknologi som gjør at endringer i koden umiddelbart speiles til nettleser eller lignende.
Discord	Verktøy for kommunikasjon og videomøter.
Messenger	Verktøy for kommunikasjon.

npm	Node Package Manager. Verktøy som benyttes for å legge til blant annet biblioteker som Vuetify.
Endepunkt	Et punkt for kommunikasjon mot backend, gjerne via et API. Backend-løsninger tilbyr endepunkt til frontend.
Responsive, responsiveness	Omhandler tilpasningsevnen til sluttbrukers skjermstørrelse. Det at innholdet tilpasser seg en mindre eller større skjerm, helt automatisk.