Høgskulen på Vestlandet

BACHELOROPPGAVE

Teleprompter

Bendik Simonsen, Jon Vollset og Thomas Valvik Yttri

Dataingeniør
Fakultet for ingeniør- og naturvitenskap
Institutt for datateknologi, elektroteknologi og realfag
Veileder: Richard Kjepso
Innleveringsdato: 8. juni 2021

Høgskulen på Vestlandet

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

TITTELSIDE FOR HOVEDPROSJEKT

Rapportens tittel: Teleprompter	*Dato:* *03.06.2021*
Forfatter(e): Bendik Simonsen, Jon Vollset og Thomas Valvik Yttri	*Antall sider u/vedlegg: 25* *Antall sider vedlegg: 28*
Studieretning: Dataingeniør	*Antall disketter/CD-er:* Ingen
Kontaktperson ved studieretning: Richard Kjepso	*Gradering:* Ingen
Merknader:	

Oppdragsgiver: Vizrt Group AS	*Oppdragsgivers referanse:* Ingen
Oppdragsgivers kontaktperson: Torbjørn Bøen	*Telefon:* 41528781

Sammendrag:

Denne rapporten beskriver utviklingsprosessen av en teleprompter webapplikasjon for Vizrt Group AS. Applikasjonen vil hente ut manus fra Viz Mosart, et studio-automasjonssystem for produksjon av nyhetssendinger.

Rapporten beskriver utviklingen av prosjektet, drøftinger og evaluering opp mot problemstillingen.

Summary:

This rapport describes the process of developing a web application regarding a teleprompter for Vizrt Group AS. The application collects the script from Viz Mozart, which is a studio-automationsystem for production of newscasts.

The rapport describes the project development, as well as discussions and evaluations regarding the approach to the problem.

Stikkord:

Blazor	SignalR	Microsoft .NET

Høgskulen på Vestlandet, Fakultet for ingeniør- og naturvitskap

Postadresse: Postboks 7030, 5020 BERGEN Besøksadresse: Inndalsveien 28, Bergen

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

FORORD

Denne rapporten dokumenterer bachelorprosjektet Teleprompter. Rapporten beskriver prosessen ved å utvikle et API og en webapplikasjon for Vizrt Group AS sitt verktøy, Mosart. Prosjektet er gjennomført av Bendik Simonsen, Jon Vollset og Thomas Valvik Yttri.

En stor takk til Vizrt Group AS som har latt oss være en del av dette spennende prosjektet. Vi ønsker spesielt å takke vår eksterne veileder Torbjørn Bøen som har veiledet og hjulpet oss gjennom hele prosjektperioden.

Vi ønsker også å takke høgskolelektor Richard Kjepso ved Høgskolen på Vestlandet som har kommet med ekspertise og veiledning gjennom hele prosjektet.

INNHOLDSFORTEGNELSE

1 INNLEDNING

Nyheter i dag kan aldri komme ut til folket fort nok. I dagens teknologiske verden kan journalister formidle nyheter fra hele kloden, og det er derfor viktig å ha et verktøy som kan benyttes på de fleste enheter tilgjengelig. Per dags dato har ikke Vizrt, et av verdens ledende selskaper innen nyhetsproduksjon, denne type verktøy i form av en webapplikasjon. Denne webapplikasjonen gir nyhetsankeret mulighet til å endre bakgrunn- og tekstfarge, i tillegg til tekststørrelse- og tempo. Den gir admin mulighet til å endre tekst når webapplikasjonen allerede kjører og nyhetsankeret er i gang med nyhetssendingen fordi nyheter fort kan endre seg og det er da viktig for en nyhetskanal å formidle tidsriktige opplysninger.

I dette kapittelet går teamet gjennom motivasjonen og målet bak prosjektet, kontekst, avgrensninger, ressurser og hvordan rapporten er bygget opp.

1.1 Motivasjon og mål

Viz Mosart er et verdensledende studio-automasjonssystem for produksjon av nyhetssendinger for kunder over hele verden. Under en slik nyhetssending er nyhetsankeret helt avhengig av teksten i teleprompteren. En teleprompter er en elektronisk enhet som viser tekst og signaler til en nyhetsanker. (Hva er en teleprompter?, 2021) Nyhetsankeret leser teksten under sending og scroller selv teksten etter ankerets eget tempo.

Figur 1 - Vizrt Logo (Vizrt, 2021)

Viz Mosart har allerede funksjonalitet for å hente ut denne informasjonen, men mangler en real-time Web Applikasjon og API for å vise og bruke dette på en brukervennlig måte med all nødvendig informasjon. Web applikasjonen skal kommunisere med et REST-basert API, basert på .NET Core Web API.

Teamets fokus vil være å dekke selskapets og dets ankerets behov for en brukervennlig Web Applikasjon som de også kan bruke i felten. Nyheter i dag kan aldri komme ut til folket fort nok, derfor er teamets mål å effektivisere og forbedre denne prosessen.

Når man har nådd det nåværende målet, har teamet flere delmål der man ønsker å forbedre applikasjonen med diverse funksjonalitet:

- Lage et API med teleprompter-applikasjon som fungerer som beskrevet i oppgavetekst
- Applikasjonen skal være responsiv og bla i tekst på brukerinput.
- Tilpasse hastighet på tekst basert på det individuelle nyhetsankeret.
- Legge til valg i innstillinger, som å bytte fra hvit tekst på svart bakgrunn til svart tekst på hvit bakgrunn.
- Optimalisere applikasjonen for ulike skjermstørrelser.

Kan en lett webapplikasjon erstatte den tradisjonelle teleprompteren i tillegg til å effektivisere arbeidsdagen til et nyhetsanker?

1.2 Kontekst

Viz Mosart er et verdensledende studio-automasjonssystem for produksjon av nyhetssendinger for kunder over hele verden. Under en slik nyhetssending er nyhetsankeret helt avhengig av teksten i teleprompteren. Nyhetsankeret leser teksten under sending og scroller selv teksten etter ankerets eget tempo.

Viz Mosart har allerede funksjonalitet for å hente ut denne informasjonen, men mangler en real-time Web Applikasjon og API for å vise og bruke dette på en brukervennlig måte med all nødvendig informasjon. Web applikasjonen skal kommunisere med et REST-basert API, basert på .NET Core Web API.

1.3 Avgrensninger

Teamet tar sikte på å videreutvikle Vizrts kode som er tildelt, hvor gruppen kan hente ut informasjon direkte ut ifra hvilket kamera som er i fokus. Ettersom kodebiten er skrevet av andre er teamet, kan det oppstå problemer i forbindelse med forståelse av koden.

Et annet problem er at teamet får hentet ut absolutt all informasjon, ikke bare teksten fra nyhetssendingen som gruppen trenger. Teamet må derfor utvikle en løsning for å hente ut relevant informasjon.

Per dags dato får teamet kun hentet ut informasjon fra den forrige nyhetssendingen først etter den nye nyhetssendingen har startet. Dette problemet har selvfølgelig store konsekvenser for en applikasjon som skal hjelpe nyhetsankeret å få opp riktig tekst til riktig nyhetssending. Det vil også legge et hinder på å oppdatere teksten i teleprompteren samtidig som den ruller kamera.

1.4 Ressurser

Ressurser gruppen har tilgjengelig, utenom egen kunnskap, er kompetanse fra Vizrt som er oppdragsgiver. Oppdragsgiver har gitt gruppen kildekode som gir tilgang til informasjonen gruppen behøver fra direkte fra deres systemer. Andre sentrale elementer er dokumentasjon fra Blazor og .NET Core som teamet benytter for å lage API-et. Blazor er Microsoft sin nyeste utviklings WEB teknologi, og er noe oppdragsgiver og intern veileder har erfaring med.

Gruppen tar sikte på å basere testingen sin på test pyramiden, hvor Unit-testing er fundamentet og bygger så videre med integrasjonstest og til slutt UI-test. En slik fordeling sparer gruppen for tid og konsekvenser av å ikke teste elementære prosesser får man går videre til mer tidkrevende og spesifikke UI-tester.

Teamet har valgt å bruke smidige metoder og Scrum der gruppen bruker Monday. Andre verktøy til kommunikasjon som: Zoom, Messenger og Teams. Bruker Blogger fra Google for å gi oppdatering om status i prosjektet. Bruker Git som versjonskontrollsystem, og GitLab som Git repository manager. For å skrive bacheloroppgaven bruker gruppen Google Docs som skriveverktøy.

Teamet ser for seg å få info og veiledning fra både intern og ekstern veileder. Begge har erfaring med utvikling, og kan hjelpe gruppen med tekniske spørsmål og veiledning. Et bra samarbeid med eksterne veiledere ser teamet på som kritisk for å lage et produkt alle er fornøyd med. Teamets fokus vil være å jevnlig vise fremgangen i prosjektet, slik at både ekstern- og intern veileder er innforstått med tankene til teamet og hva teamet ser for seg i fremtiden.

1.5 Oppbygging av rapporten

Kapittel 1 beskriver målet med prosjektet, samtidig som man blir kjent med oppdragsgiver. I dette kapittelet får man i tillegg kjennskap til prosjektets avgrensninger.

Kapittel 2 går mer i dybden på prosjekteier og dens bakgrunn. Her diskuteres tidligere arbeid, initielle krav- og løsning til prosjektet, som setter standard og rammer for det endelige resultatet.

I kapittel 3 framheves alternative løsninger for prosjektet i startfasen. Disse tilnærmingene diskuteres og ses i sammenheng med hverandre, hvor løsningen teamet har valgt er resultatet av denne diskusjonen. Løsningen teamet går for blir fremhevet og det gis en forklaring til hvorfor nettopp denne løsningen ble valgt.

Videre formidles det om de forskjellige verktøyene teamet bruker i prosjektet og hvorfor nettopp disse ble valgt. Sist gjennomgås prosjektmetodikk og evalueringsplan hvor utvikling, prosjektplan og risikovurdering står i fokus.

Kapittel 4 diskuterer i en mer detaljert måte tilnærmingene teamet tok i henhold til design og arkitektur av prosjektet og applikasjonen.

Kapittel 5 beskriver evalueringsmetodene som teamet har benyttet og resultatet som følge av bruken av disse.

Kapittel 6 tar for seg og gjennomgår resultatene fra prosjektet. Dette kapittelet inneholder konsekvenser av teamets tilnærming til prosjektet og hvordan handlingene påvirket det endelige resultatet.

I kapittel 7 diskuteres det hva teamet kunne gjort annerledes om man kunne utført prosjektet på nytt. Hvilke erfaringer teamet har gjort seg og ytterligere diskusjon står i sentrum.

I kapittel 8 skriver teamet et sammendrag og diskuterer konklusjonen av prosjektet, og hvordan prosjektet kan utvikles videre fra nåværende ståsted.

Kapittel 9 inneholder alle teamets referanser som er benyttet i prosjektet.

Kapittel 10 inneholder en visualisering av risikovurdering, hvilke hendelser som kan inntreffe og dens alvorlighets- og påvirkningsgrad for prosjektet. I tillegg legges det ved et GANTT diagram som visualiserer arbeidsmetodikk og den overordnede tidsplanen for prosjektet, i tillegg til en brukermanual for applikasjonen.

2 PROSJEKBESKRIVELSE

Dette kapittelet beskriver utviklingen av en webapplikasjon med utgangspunkt i Vizrt sitt system for nyhetssendinger. Bakgrunnen for applikasjonen er å gjøre Vizrt- og nyhetsankere sin hverdag enklere ved å kunne benytte samme applikasjon for å lese og endre tekst, justere skriftstørrelse og tekstfart.

2.1 Praktisk bakgrunn

2.1.1 Prosjekteier

Vizrt er en verdensledende leverandør av visuelle verktøy innenfor grafikk og videoredigering til den digitale mediebransjen. Vizrt ble startet i 1997 og har per dags dato over 700 ansatte i en rekke land. (About Vizrt, 2021)

2.1.2 Tidligere arbeid

Vizrt har ikke arbeidet med et prosjekt for en teleprompterapplikasjon tidligere. Teamet starter med andre ord helt fra scratch, men har fått tildelt kode som gjør det mulig å snakke med Vizrts systemer for å hente ut informasjonen gruppen trenger.

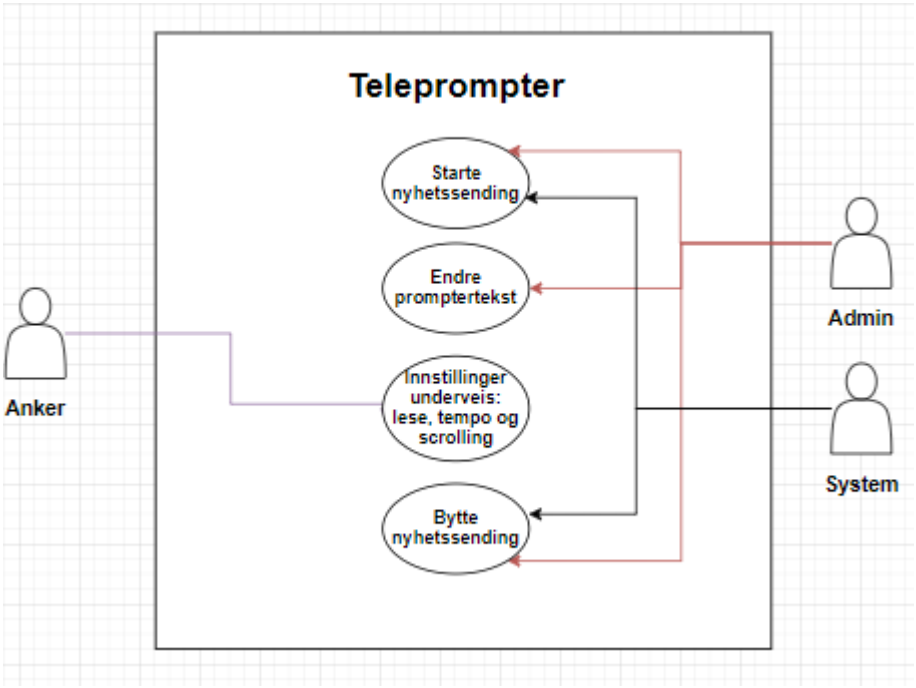
2.1.3 Initielle krav

Teamet og Vizrt hadde tidlig en gjennomgående god samtale hvor de forskjellige tankene og rammene rundt prosjektet ble diskutert. Oppdragsgiver beskrev et konsept hvor ankeret og de som sitter i bakgrunnen under en nyhetssending skal kunne lese, få oversikt og endre teksten som vises i teleprompteren.

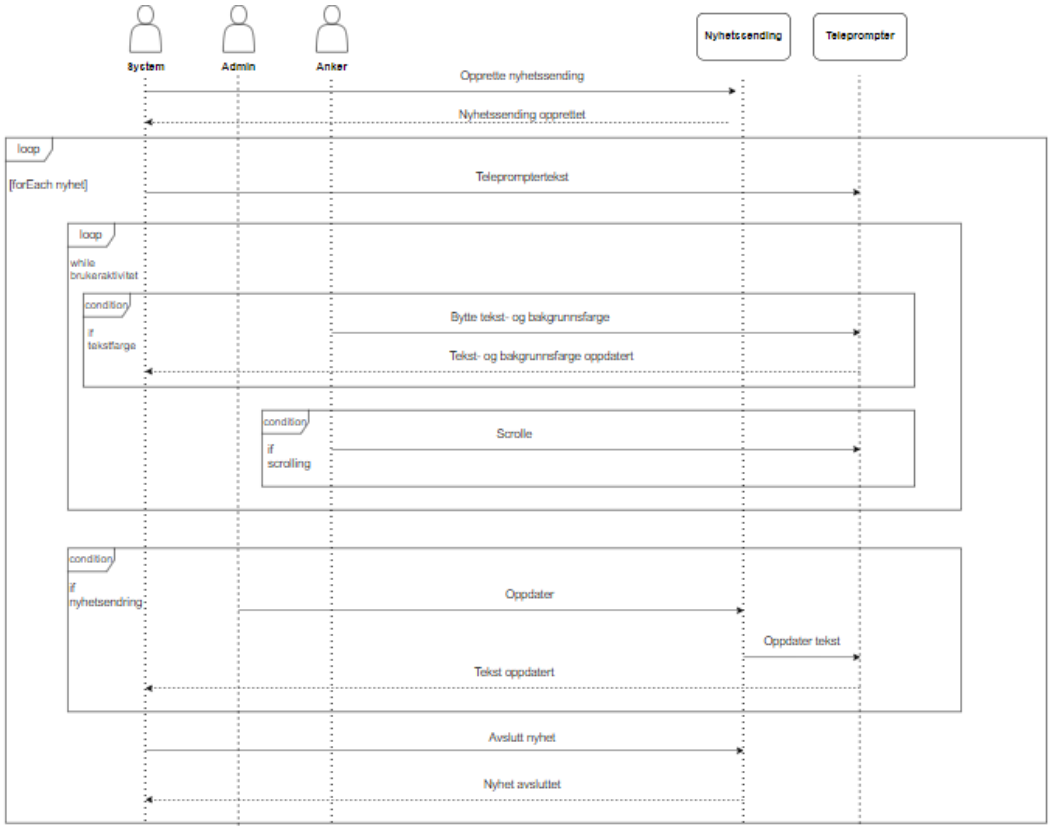
Her er kravene listet opp punktvis:

- En fungerende webapplikasjon
- App-en må kommunisere med et REST-basert API, basert på .NET
- Innstillinger for ankeret, som å justere tempo og scrolle teksten
- Vise rett tekst i teleprompteren avhengig av handlinger gjort av admin/anker

Figur 1 viser et brukstilfellediagram av hvordan teamet ser for seg at systemet skal fungere. Både system og admin vil kunne starte en nyhetssending, og admin vil i tillegg kunne ha mulighet til å endre teleprompterteksten underveis ettersom nyheter fort kan endres. Ankeret, som i og for seg er det viktigste leddet med utgangspunkt i oppgaven, skal lese teksten og gjøre endringer underveis basert på behov. Om sendingen går helt til slutt vil systemet bytte til neste sending, men admin vil også kunne ha mulighet til å bytte sending basert på hendelser som plutselig kan inntreffe.



Figur 2 - Viser brukstilfellediagram



Figur 3 - Viser sekvensdiagram

2.1.4 Initiell løsnings-idé

Målet med oppgaven er å ha en så enkel designløsning som overhodet mulig, da man i nyhetsverdenen fort må omstille seg. Et enkelt design gjør nettopp det lettere for de i administrasjonen. Ankeret skal få opp riktig tekst til riktig nyhetssending på riktig kamera, og teksten skal være oversiktlig med gode kontraster slik at det ikke skal være tvil om hva som står i teleprompteren. Ankeret skal kunne scrolle og endre tempo etter behov. Administrasjonen bak skal ha mulighet til å endre teksten midt i nyhetssendingen, om de ser behov for å legge til eller fjerne tekst.

3 DESIGN AV PROSJEKTET

I dette kapittelet utforskes forskjellige tilnærminger til løsning og hvilken teamet anser som den best mulige løsningen. I tillegg vil emner som valg av verktøy, prosjektmetodikk og evalueringsplan bli sett nærmere på.

3.1 Forslag til løsning

Det finnes flere gode løsninger for å oppfylle kravene til oppdragsgiver. Felles for alle er et API utviklet i .NET 5.0. Dette er fordi programvaren til oppdragsgiver er basert på .NET, og derfor naturlig at teamet benytter det samme utviklingsrammeverket når gruppen skal lage APIet. Det vil med andre ord si at forskjellen mellom løsningene er rammeverk og teknologien gruppen bruker for å utvikle webapplikasjonen.

3.1.1 Alternativ løsning 1

Gruppen vurderte først en løsning der webapplikasjonen blir utviklet med JavaScript, og React som mulig rammeverk. Dette er en løsning som baserer seg på svært godt dokumenterte og ressurssterke teknologier. JavaScript med React som rammeverk er teknologi gruppen har god kjennskap til.

3.1.2 Alternativ løsning 2

Et annet alternativ gruppen vurderte var å utvikle webapplikasjonen med et hybrid rammeverk. Fordelen med et hybrid rammeverk er at gruppen produserer en applikasjon for hver plattform. Dette vil typisk være separate applikasjoner for web, iOS og Android. De aktuelle rammeverkene gruppen har vurdert er Flutter og Ionic, som er godt dokumenterte teknologier.

3.1.3 Alternativ løsning 3

Alternativ 3 er en løsning der gruppen benytter C# som utviklingsspråk i både front- og back-end. ASP.NET Blazor er en nyere, men svært populær måte å utvikle webapplikasjoner i .NET stacken.

3.1.4 Diskusjon av alternativene

JavaScript er den løsningen gruppen har mest erfaring med, og vil dermed være et tryggere alternativ. Dette vil føre til at gruppen tidligere kan komme i gang med iterasjonene, noe som igjen vil føre til at en sparer tid på opplæring. Ved å velge JavaScript løsningen kan gruppen benytte rammeverket React, som det er stor interesse for blant gruppemedlemmene.

Kryss-plattform rammeverket gir en stor fordel med at teamet får produsert flere applikasjoner fra samme prosjekt. Dette fører ofte til bedre optimaliserte løsninger for de ulike enhetene sluttbrukerne benytter. Det er dog ikke et krav fra oppdragsgiver å produsere mobilapplikasjoner. Prosjektet kommer til å ha en simpel front-end, som gjør at fordelen med optimalisering faller litt bort. Dette blir derfor vurdert som den løsningen som stiller svakest sammenlignet med de andre.

Fordelen med Blazor er at utviklerne kan bruke ett programmeringsspråk. Det gir også et stort læringsutbytte ettersom ingen av utviklerne har tidligere erfaring med Blazor, .NET eller C# generelt. Ved å gå for Blazor kan prosjektet teste både front- og back-end med samme rammeverk, og en kan levere et produkt utviklet med teknologi oppdragsgiver har god kjennskap til.

3.2 Valgt løsning

Hovedfokuset til gruppen er er å komme i mål med all nødvendig funksjonalitet, og levere et produkt oppdragsgiver kan bruke videre. Ettersom at Vizrt hovedsakelig benytter .NET i Mosart, konkluderte gruppen med at en webapplikasjon basert på C# rammeverket Blazor var den logiske løsningen. Denne løsningen gir utviklerne muligheten til å sette seg inn i ny teknologi uten forkunnskaper om C#, noe som gir et stort læringsutbytte.

HVL veileder og oppdragsgiver har god erfaring med både Blazor og .NET, noe som naturligvis blir en stor ressurs for gruppen.

3.3 Valg av verktøy

I dette kapittelet vil det bli presentert ulike verktøy som gruppen har brukt. I tillegg ulike språk, bibliotek, rammeverk, og kommunikasjons- og planleggingsverktøy.

3.3.1 Verktøy

Visual Studio

Microsoft Visual Studio er et integrert utviklingsmiljø for Microsofts .NET plattform og passer som hånd i hanske til prosjektoppgaven hvor teamet skal utvikle en webapplikasjon for Windows. Visual Studio har gode testegenskaper, er standardisert og er et “easy-to-use” grensesnitt.

Gitlab

GitLab er en webbasert “hosting” tjeneste for versjonskontroll ved bruk av Git. Prosjektet lagres i GitLab hvor teamet kan ha kontroll på forskjellige grener eller “branches” i tillegg til eventuelle feil. GitLab er også brukt av teamets prosjekteier og er den enkleste måten å holde alle parter oppdatert på koden.

3.3.2 Språk, bibliotek og rammeverk

C#

.NET utviklingsmiljøet bruker C# som utviklingsspråk, derav gruppens utviklingsspråk. C# er et moderne objektorientert språk, som lar utvikleren lage sikre og solide applikasjoner. (A Tour of C# - C# Guide, 2021)

Blazor

Gruppen valgte å bruke det .NET baserte rammeverket Blazor for å utvikle webapplikasjonen. Blazor gjør det mulig å utvikle komponent-baserte web UI i C# i stedet for JavaScript. Rammeverket gir og muligheten for å kjøre webapplikasjonen på WebAssembly i tillegg til server. (Blazor | Build client web apps with C# | .NET, 2021)

Swagger

Gruppen har benyttet Swagger Inspector for å teste API endepunkt. Swagger tilbyr et spekter av løsninger, der Inspector lar utvikleren inspisere og teste API request og responses, og bekrefte at de fungerer som forventet. (API Testing, 2021)

SignalR

SignalR er et bibliotek for å gi webapplikasjonen «real-time» funksjon i Microsoft .NET. Ved bruk av SignalR kan teamet lage en webapplikasjon som krever høyfrekvensoppdateringer fra serveren. (SignalR | What, Why and How About SignalR | 2017). Hver gang det forekommer endringer på serversiden, som for eksempel endring i nyhetssending, vil klienten umiddelbart få vite dette og prosessere informasjonen.

3.3.3 Kommunikasjon- og planleggingsverktøy

Monday

Teamet bruker Monday som planleggingsverktøy. Monday er utrolig enkelt å bruke og hjelper gruppen med oversikt over prosjektet, så vel som å overholde diverse innleveringer og deadlines. I tillegg bruker teamet Monday for å skrive notater, for eksempel fra møter med veileder, og å holde tritt på antall timer som blir brukt til prosjektet.

Teams

Gruppen bruker Teams for å kommunisere med oppdragsgiver, som er et verktøy for kommunikasjon og samarbeid på tvers av grupper så vel som enkeltpersoner. I Teams har gruppen opprettet en egen kanal hvor man kan spørre spørsmål og utføre ukentlige digitale møter.

Google Docs

Teamet benytter Docs for å skrive prosjektet, da alle i gruppen kan jobbe samtidig på oppgaven. Google Docs er et gratis verktøy som brukes for å lage og dele dokumenter. Man får tilgang til en teksteditor som minner mye om Word, som de fleste er godt kjent med. Fordelen med å bruke et slikt verktøy er at man lettere kan samarbeide og planlegge, i tillegg til at dokumentet lagrer automatisk når man skriver, slik at man slipper å engste seg for å miste det man har jobbet med.

Zoom

Zoom er et verktøy for videomøter og konferanser som brukes til interne møter innad i teamet. Zoom har gruppen blitt veldig godt kjent med gjennom digital undervisning siden pandemiens start i mars 2020 og er det foretrukne kommuniseringsverktøyet.

Blogger

Teamet benytter Blogger for å legge ut blogginnlegg om prosjektet. Blogger er Googles bloggtjeneste som er enkel, praktisk og gratis å ta i bruk. Teamet forsøker i den grad det lar seg gjøre å benytte seg av Googles produkter.

3.4 Prosjektmetodikk

3.4.1 Utviklingsmetodikk

Teamet baserer seg på smidig utvikling. Metodikken gjør at gruppen tidlig kan identifisere risiko og utbedre løsninger. Testing av kode er et viktig fokusområde i smidig utvikling. I hver iterasjon vil testing av kode være nøkkelen til god kjørbare kode. Dermed kan teamet jobbe seg ferdig med en iterasjon og vite at den er kjørbar. Dette bidrar til mindre risiko på slutten av prosjektet. (Prosjektstyring og smidig utviklingsmetodikk | Digitaliseringsdirektoratet, 2021

Gruppen bruker utviklingsmetodikken Scrum. Dette er en metode som ligger under smidig utvikling. Scrumverktøyet som blir brukt er Monday, dette verktøyet er beskrevet i kapittel 3.3.3.1. Scrum er et rammeverk basert på utvikling og leveranser i korte iterasjoner med en fast lengde. Oppgavene blir også prioritert fra hva som er viktigst for prosjektet og kunden. I Scrum finnes det tre roller, Produkteier, Scrum team og Scrum master. Produkteier eier prosjektet. Teamet skal utføre oppgavene på sin måte. Scrum master skal ha god kunnskap om Scrum og optimalisere utviklingen. (Hva er egentlig Scrum?, 2021) Teamet bruker en egen tilnærming av Scrum som passer bedre gruppens arbeidsmåte.

3.4.2 Prosjektplan

For å illustrere teamets fremdriftsplan er det benyttet Gantt skjema. Gruppen valgte å bruke et Gantt skjema fordi det gir en ryddig og god oversikt for den overordnede planen, samtidig som det kaster lys på viktige delmål og frister. Denne løsningen, sammen med Scrum baserte Monday, gjør at gruppen til enhver tid vet hvilke oppgaver som skal gjøres, og de fristene som haster mest. Det er også viktig å påpeke at hvert delmål bygger videre på det forrige delmålet. For eksempel vil utkast 2 først begynne når teamet er ferdig med utkast 1.

Forarbeid prosjekt

Forarbeid prosjektperioden inneholder innleveringer i regi av HVL. Her er Mål og metode innlevering 1 og 2, forprosjektrapport og forprosjekt presentasjon. Gruppen har også jobbet med et Blazor testprosjekt i denne perioden, for å bli kjent med teknologien før selve prosjektet. Forarbeid prosjektperioden varer i tidsrommet 23.03 – 23.04.

Baseprosjekt perioden

Baseprosjekt perioden inneholder de forskjellige iterasjonene. I hver iterasjon er det tilegnet tid for testing og implementering, samt middels grad av optimalisering. Baseprosjekt perioden varer i tidsrommet 19.04-14.05.

I løpet av iterasjon 1 skal de grunnleggende funksjonene i APIet være på plass, samtidig som at gruppen skal ha en webapplikasjon som kan kalle på APIet. Denne oppgaven er en stor del av prosjektet, og er svært nødvendig for videre utvikling av Teleprompter. Iterasjon 1 starter 19.04 og avsluttes 23.04.

Iterasjon 2 handler om å bruke dataen fra APIet, og formidle denne til brukeren. Her er webapplikasjonen i fokus. Webapplikasjonen viser tekst på en funksjonell måte, og nødvendig funksjonalitet skal implementeres. Iterasjon 2 starter 26.04 og avsluttes 30.04

Iterasjon 3 inneholder optimalisering av både API og webapplikasjon. Her skal gruppen gå gjennom prosjektet og se om det kan gjøres forbedringer, og utbedre webapplikasjon med tanke på brukervennlighet, funksjonalitet og responsivitet. Iterasjon 3 begynner 3.05 og avsluttes 7.05.

Finjusteringer og ekstra tid blokken er satt av i tilfelle gruppen trenger mer tid på optimalisering, møter på uventede problemer eller ønsker å videre utbedre webapplikasjon. Denne tiden kan også bli brukt til brukertesting dersom det blir muligheter for dette. Starter 10.05 og avsluttes 14.05.

Ekstra funksjonalitet

I denne perioden skal gruppen implementere ekstra funksjonalitet som sammen med oppdragsgiver blir vurdert som nyttig. Ekstra funksjonalitet perioden varer i tidsrommet 17.05 – 31.05.

Rapport

I blokken Rapport skal gruppen bruke tid på å utforme rapporten som skal leveres i slutten av prosjektet. Det er viktige milepæler her, som Utkast 1 25.05, utkast 2 2.06 og endelig rapport innlevering 8.06. Blokken spres fra start baseprosjekt grunnet kontinuerlig skriving underveis. Det skal også nevnes at Forprosjekt rapport gir gruppen et godt utgangspunkt til rapporten, og kan betraktes som Utkast 0. Rapport perioden varer i tidsrommet 19.04 – 08.06.

3.4.3 Risikovurdering

En risikoanalyse utføres for å avdekke risiko i alle faser av prosjektet hvor informasjonen legger grunnlaget for de beslutningene som tas. I risikoanalysen har teamet forsøkt å besvare tre sentrale spørsmål; hva kan gå galt, hva er sannsynligheten for at de uønskede hendelsene inntreffer og hvilke konsekvenser de uønskede hendelsene kan medføre.

Risikoen måles fra 1-5 hvor 1 er lav sannsynlighet og liten påvirkningsgrad, mens 5 er høy sannsynlighet og stor påvirkningsgrad. Alvorlighet er sannsynlighet multiplisert med påvirkningsgrad.

3.5 Evalueringsplan

Når prosjektet er ferdig og produktet er klar for test, vil oppdragsgiver utføre denne testingen på sine nyhetsankere. På den måten vil man se om produktet er godt nok til å kunne tas i bruk. Grunnen til at man ikke tester ut mot et større publikum er at prosessen bak det å få teleprompterteksten er såpass omfattende, med tanke på at man må ha tilgang til Vizrts systemer og en teleprompter.

Å teste systemet ut mot nyhetsankere vil være en god løsning da de allerede er kjent med lignende system og vet hva de ønsker og ikke ønsker. Helt til slutt vil teamet ta for seg tilbakemeldingene fra oppdragsgiver og man vil da kunne ha et klarere mål om framtidig løsning.

4 Detaljert design

I utviklingen av en webapplikasjon er det viktig å se på arkitektur og oppbygningen av prosjektet. I dette kapittelet skal teamet gå nærmere inn de forskjellige aspektene ved prosjektdesignet og forklare disse i detalj.

4.1.1 UI

Gruppen kom frem til at det mest effektive ville være å lage én applikasjon som kunne benyttes på pc, mobil og tablet. Teamets største fokusområde er å lage et godt og forståelig UI som er responsiv på flere plattformer, da nyhetsankeret nødvendigvis ikke alltid er i studio med alt av utstyr. Nyhetsankeret skal på kort tid kunne få opp applikasjonen uavhengig av plattform for å kunne formidle nyhetene som de skjer.

For at applikasjonen skal være så responsiv som mulig, vil skalering være et viktig element. Ettersom de fleste plattformer og enheter i dag kommer i forskjellige størrelser, vil det være ytterst viktig å kunne skalere tekst og knapper i direkte tilknytning til enhetens størrelse.

4.1.2 Hovedside

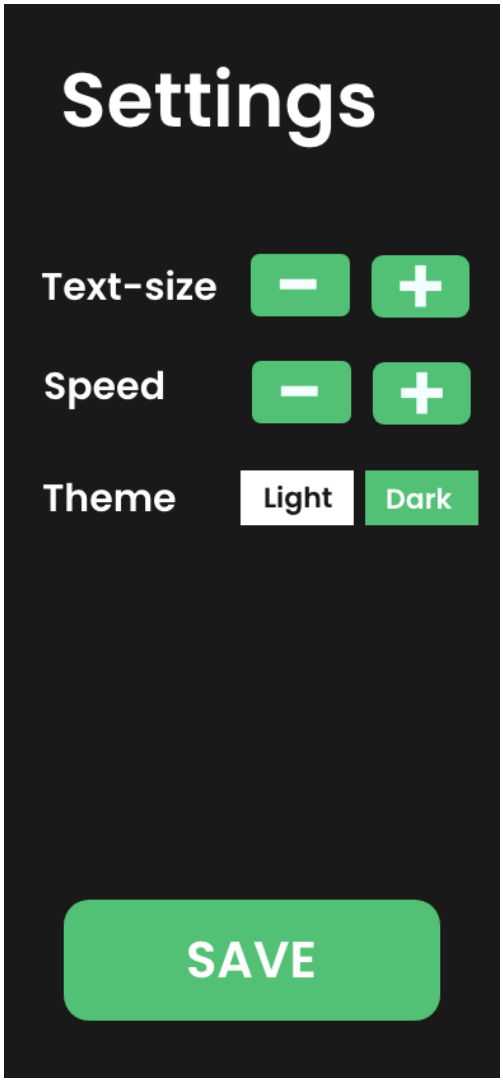
Hovedsiden til webapplikasjonen skal gjøres så enkel og oversiktlig som mulig. Det er denne siden nyhetsankeret ser når man leser tekst fra teleprompteren og det gjør egenskaper som teksttype og kontrast ekstremt viktig. Teamet ser for seg tekst på store deler av siden, hvor man har et lite tannhjul oppe i høyre hjørne hvor nyhetsankeret sømløst skal kunne endre tekst- og bakgrunnsfarge eller farten teksten vises i.

Figur 4 – illustrasjon av webapplikasjonen på en pc og mobiltelefon

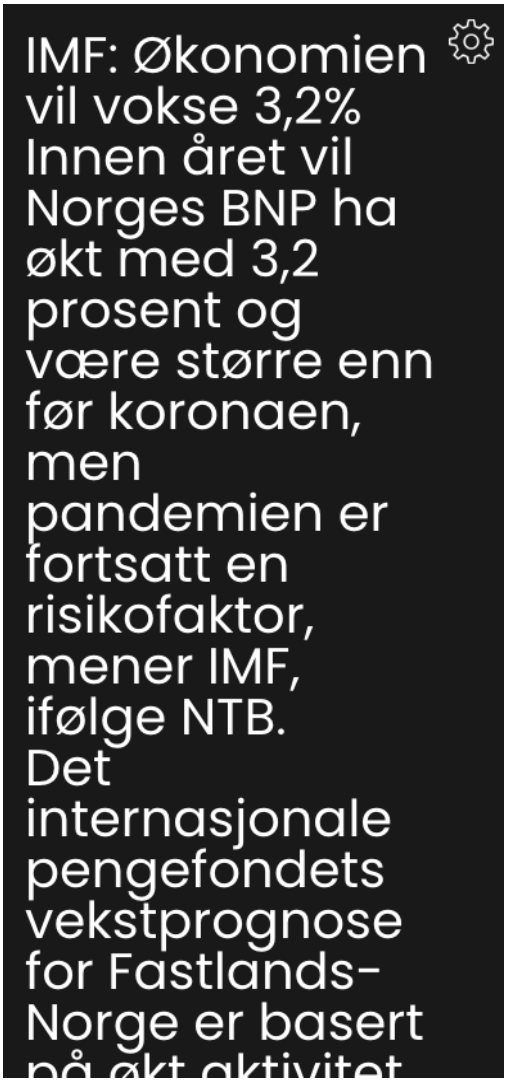
Som illustrasjonen over viser, har teamet fokusert på et veldig enkelt design for å gjøre jobben til nyhetsankeret enklere og mer håndterbart, spesielt i feltarbeid.

4.1.3 Innstillinger

Som nevnt i punkt «**4.1.2 Hovedside**» ser teamet for seg et tannhjul øverst til høyre på hovedsiden. Her kan nyhetsankeret bytte tekst- og bakgrunnsfarge (fra hvit tekst og svart bakgrunn til svart tekst og hvit bakgrunn), justere farten teksten scrolles i og tekststørrelse. Utgangspunktet til denne farten henter teamet direkte fra informasjon registrert på hvert enkelt nyhetsanker, men man skal da i tillegg ha mulighet til å endre denne selv. I tillegg vil brukeren kunne gå i fullskjermmodus.



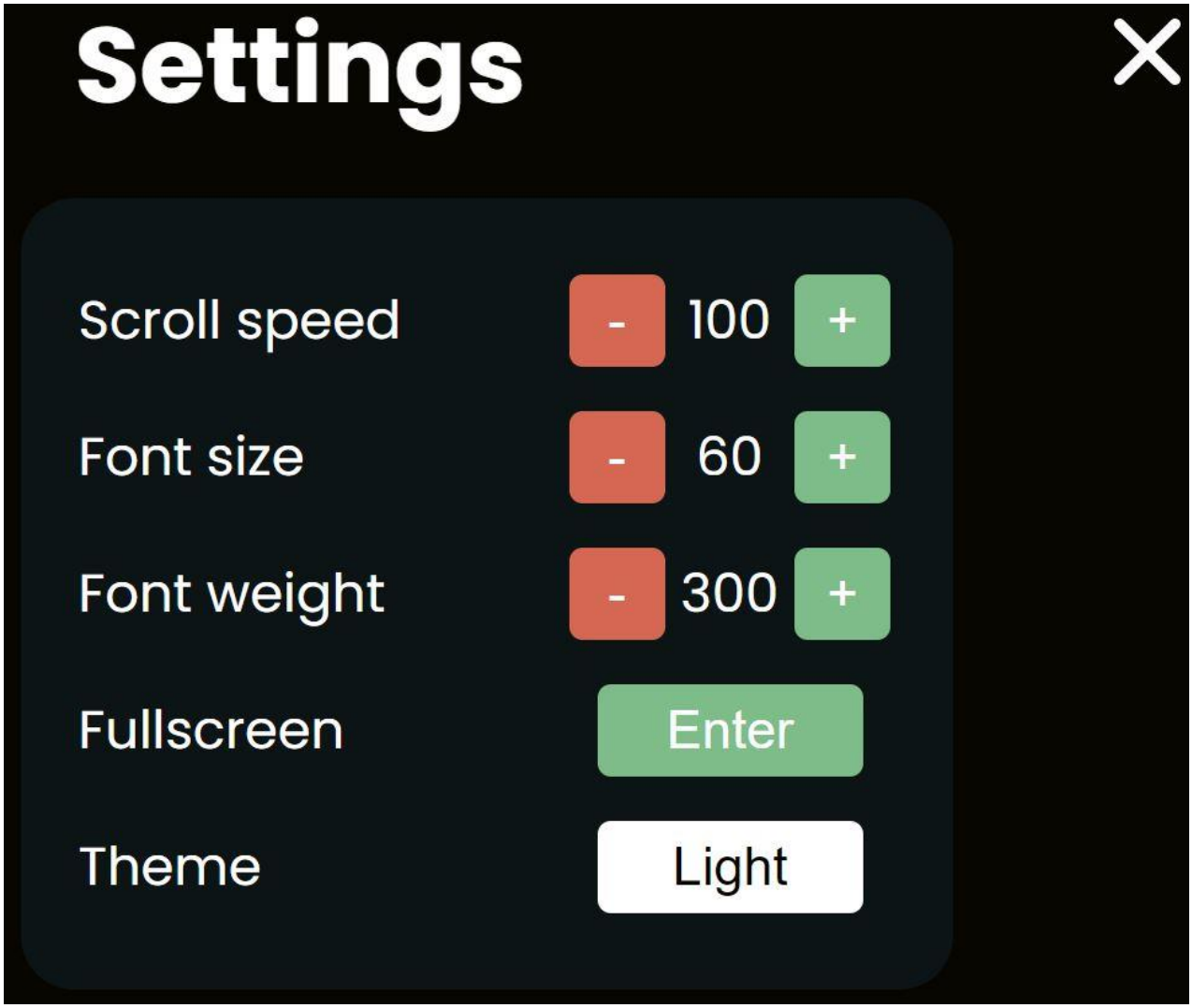
Figur 5 – Mulig løsning for mobil



Figur 6 – Hovedside på mobil

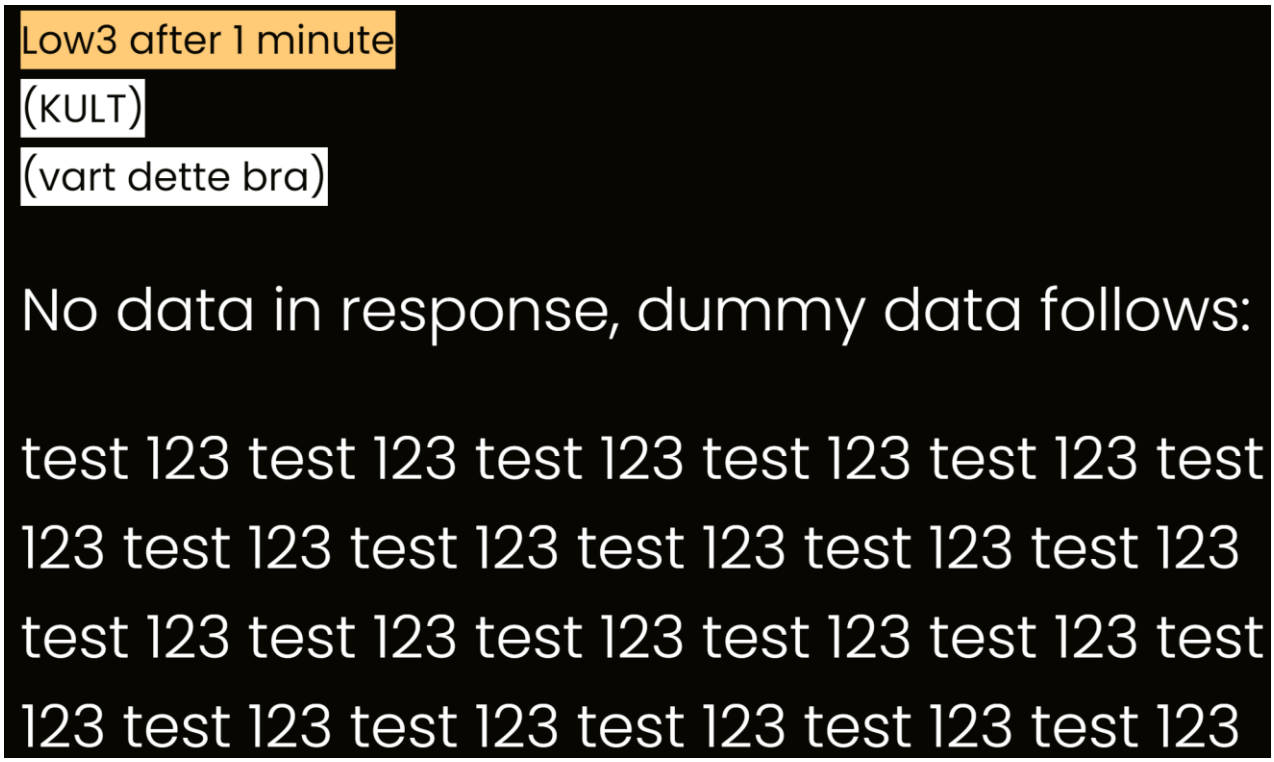
Teamet ser for seg to knapper der nyhetsankeret justerer farten enten ved klikk av venstre knapp for en saktere visning, eller til høyre for en raskere visning. Måleenheten baserer seg på antall leste ord i minuttet og vil kunne justeres fra 75-220 ord per minutt.

Tannhjulets responstid vil prioriteres, slik at nyhetsankeret ikke bruker unødvendig lang tid på å justere de forskjellige egenskapene til webapplikasjonen. Publikum vil også fort merke om nyhetsankeret gjør andre ting enn å lese teksten i webapplikasjonen, så teamets fokus vil være å gjøre prosessen så sømløst som mulig.



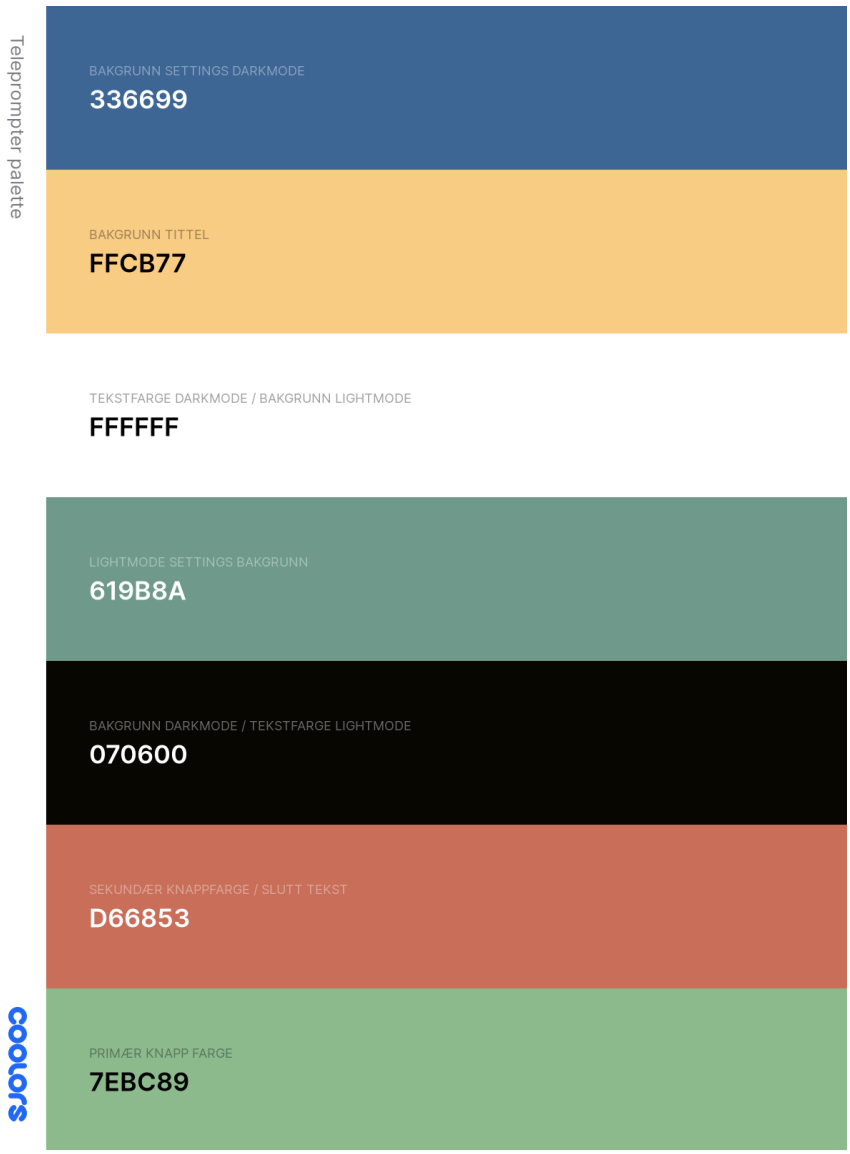
Figur 7 – Illustrasjon av innstillinger for admin i ny fane i nettleser og nåværende løsning for mobil

Når brukeren er fornøyd med valgene og trykker på krysset, vil innstillingene gjenspeile seg på promptteksten. I figur 8 er tittelen for storyen markert med en oransje bakgrunn, mens de to punktene med hvit bakgrunn er tilleggsinformasjon for den aktive storyen.



Figur 8 – Illustrasjon av hovedside av webapplikasjon i nettleser

Figur 9 viser fargekartet for prosjektet. Fargene er nøye valgt ut med tanke på kontrast og utseende, slik at teksten skal bli så lett å lese som overhodet mulig.

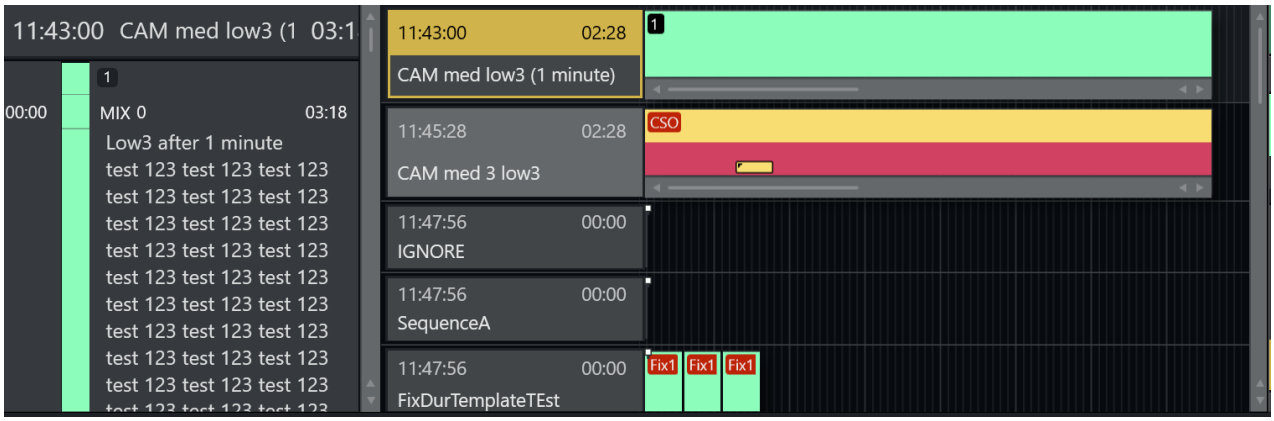


Figur 9 – Illustrasjon av fargekart i prosjektet

4.1.4 Vizrt GUI

Når Vizrt kjører sine nyhetssendinger, benytter de GUI-en som vist i figur 10. GUI-en gir selskapet oversikt over hvilken story de er på og varigheten på denne. I tillegg kan man i figur 10 til venstre se prompterteksten for storyen som kjøres og her kan Vizrt pause, endre og bytte nyhetssending som de selv ønsker.

Det er fra disse storyene teamet innhenter informasjon om ankerets navn, readrate og prompertekst. Som man kan se av figur 10, kan én story inneholde flere items. Disse itemsene kan inneholde hver sin prompertekst, og det er derfor viktig å hente ut riktig informasjon til riktig tid tilknyttet riktig story.



Figur 10 – Illustrasjon av GUI til Vizrt sine systemer

5 Evaluering

Evaluering av webapplikasjonen er et av de viktigste aspektene i prosjektet. Innspill fra prosjekteier underveis i prosjektet vektlegges og ved å ha ukentlige møter er teamet sikker på at produktet blir som alle parter har sett for seg. Siden prosjektet er en webapplikasjon som skal vise og manipulere teleprompertekst kan teamet også benytte seg av tilbakemeldinger direkte fra fokusgruppen, nemlig nyhetsankere.

5.1 Evalueringsmetode

5.1.1 Testing

Testing er en helt sentral del av prosjektet og det er benyttet flere forskjellige testverktøy underveis.

Debugger i Visual Studio

Debugger i Visual Studio gjør det mulig å sette breakpoint forskjellige steder i koden for å sjekke verdier for eksempel før man går inn i en metode. På den måten kunne teamet sjekke hvilke verdier metoder opererte med, i tillegg til om systemet i det hele tatt kjørte forskjellige metoder. Det var ved hjelp av debugging at teamet ble oppmerksom på det største problemet vårt, nemlig at metodene som var skrevet ikke fikk til å oppdatere tekst og readrate når en ny story ble satt i gang.

Swagger

Swagger er benyttet med stor suksess for å teste opp mot API-et. Selv om gruppen til slutt endte opp med en løsning som ikke benyttet selve API-et, var et bra verktøy å bruke for å sjekke at man fikk ut riktig informasjon til riktig tid. I tillegg var det fint å kunne benytte Swagger for å se at gruppen uthentet korrekte verdier fra XML og fikk til å sende dette i JSON.

Logger

Teamet har benyttet logger i Visual Studio for å få ut informasjon fra koden som kan være nyttig for å teste applikasjonen. Logger i tillegg til debugging har hjulpet gruppen til å se helheten i prosjektet og innhente informasjon om eventuelle feil i koden og systemet.

5.1.2 Tilbakemelding fra oppdragsgiver

Tilbakemeldinger fra oppdragsgiver har vært til stor hjelp for underveisevaluering av prosjektet. Ved å jobbe tett med Torbjørn Bøen har teamet fått en forståelse av hvilken løsning Vizrt ser for seg og man har da kunne gjort endringer fortløpende ut ifra deres behov.

6 Resultater

I dette kapittelet tar teamet for seg og gjennomgår resultatene fra prosjektet. I tillegg inneholder kapittelet en gjennomgang av konsekvenser av teamets tilnærming til prosjektet og hvordan handlingene påvirket det endelige resultatet.

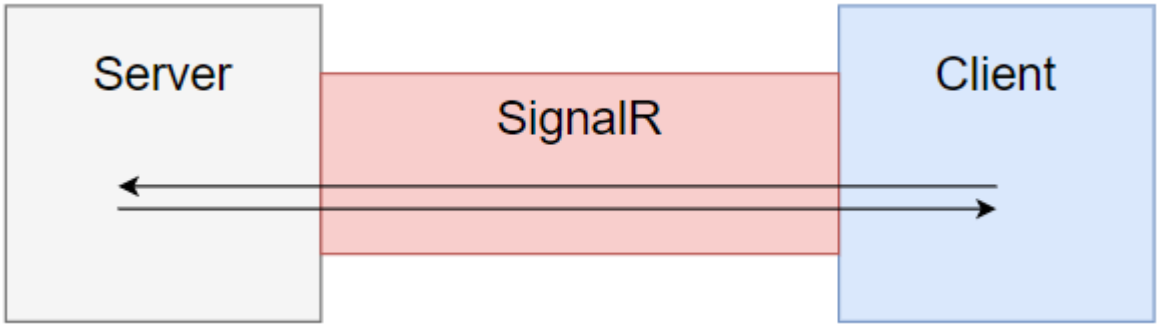
6.1.1 Design og brukervennlighet

Designet for webapplikasjonen var så godt som spikret ferdig av oppdragsgiver i innledningen til prosjektet. Vizrt visste godt hva de ville ha, og hvordan de så for seg at webapplikasjonen skulle se ut. Deres ønske var også det teamet så for seg løsningen skulle bli seende ut som, og resultatet ble et design alle parter var kjent med og kunne stille seg bak. Det som var mest tidkrevende var å uthente riktig informasjon til riktig tid. Når det var i boks, var framvisning av tekst en enkel oppgave.

Resultatet er en webapplikasjon med et enkelt og godt design hvor brukeren kan utføre diverse operasjoner på teksten. Disse funksjonene er lagt til i en enkel innstillingsmeny hvor hver av funksjonene er listet opp.

6.1.2 Backend

Etter mye arbeid med å få koblet systemet opp mot API-et for å innhente riktig informasjon ble det sent i prosjektet bestemt at man skulle benytte SignalR. På den måten slipper man å foreta konsekvente spørringer mot API-et, nå kan teamet innhente informasjonen som er nødvendig så snart systemet oppfatter en endring. Dette var en mye bedre løsning enn den gruppen opprinnelig så for seg og API-et ble fort bare et «means to an end» som ble benyttet for å sjekke at informasjonen stemte overens. Med andre ord ble API-et ikke benyttet som en del av løsningen, da resultatet med bruk av SignalR som overfører informasjonen til webapplikasjonen så snart den merker endringer ble såpass bra.



Figur 11 – Illustrasjon av SignalR og hvordan den lenker sammen back- og frontend

7 Diskusjon

I dette kapittelet diskuteres det hva teamet kunne gjort annerledes om man kunne utført prosjektet på nytt. Hvilke erfaringer teamet har gjort seg og ytterligere diskusjon står i sentrum.

7.1 Konsekvenser

7.1.1 Verktøy

Teamet har brukt en rekke forskjellige verktøy i utviklingen av denne webapplikasjonen. I starten så ikke teamet for seg å ha et eget verktøy for prosjektoversikt i form av frister, møter etc. Etter hvert så gruppen behovet og valgte nettstedet Monday. Konsekvensen av å ikke bruke et slikt verktøy for å holde oversikt tror teamet ville gjort arbeidet fryktelig tungt i det lange løp. I Monday har man referat fra møter, både med intern- og ekstern veileder, notater fra egne oppdagelser og de forskjellige tidsfristene.

Konsekvensen av å ikke skrive koden til webapplikasjonen i Microsoft .NET ville vært enorm. Ettersom hele backenden til Vizrt er skrevet i nettopp Microsoft .NET med bruk av C# ville det vært tungvint for dem dersom koden skulle bli skrevet i et annet rammeverk og språk. Men, ettersom Microsoft .NET og C# ikke er kjent i gruppen fra tidligere, har man måtte sette av store ressurser på å sette seg inn i dette, ressurser som teamet heller kunne benyttet på andre områder.

SignalR ble gruppen oppmerksom på når oppdragsgiver informerte om at de implementerte rammeverket i sine systemer foruten prosjektet. Konsekvensen av å bruke spørringer mot API i stedet for å pushe endringer i server umiddelbart til klienten(e) ville ikke vært så stor, men gruppen føler at det er en mye bedre disposisjon av ressurser at serveren kan fortelle klienten når det er endring i stedet for å foreta konsekvente spørringer hvert millisekund.

7.1.2 Arbeidsmetode

Gruppen er godt fornøyd med valget av Scrum som utviklingsmetode. Når teamet hadde korte iterasjoner i utviklingen var det lett å holde oversikt over hvor langt i prosjektet man hadde kommet i tillegg til at man visste at koden fungerte etter å ha gjennomført tester i henhold til testpyramiden.

8 Konklusjon og videre arbeid

Målet med bachelorprosjektet var å utvikle en webapplikasjon for Vizrt som kunne snakke med deres systemer og framvise data i form av tekst til deres nyhetsankere. Webapplikasjonen skulle kommunisere med et REST-basert API som skulle basere seg på Microsoft .NET Core Web API. Per dags dato er gruppen svært nærme å nå dette målet. Riktig tekst til riktig nyhetssending hentes ut problemfritt og back- og frontend er nå satt sammen.

De initielle kravene er med andre ord nådd. Gruppen ser nå fremover mot forbedringer av webapplikasjonen, dette kan være nye og bedre funksjonaliteter, tilgang til flere enheter etc. Det eneste punktet det fortsatt jobbes med er optimalisering for bruk av flere enheter, som beskrevet i nærme detaljer i punkt 8.2.2.

8.1 Nådde mål

8.1.1 Real-time webapplikasjon

Webapplikasjonen er ferdig. Gruppen har klart å hente ut relevant informasjon fra Vizrt sine systemer og framviser dette i webapplikasjonen på en god og oversiktlig måte.

I tillegg er har webapplikasjonen real-time funksjon som gjør at ankeret får lese riktig tekst til riktig nyhetssending, også når det gjøres endringer i bakgrunnen.

8.1.2 Innstillinger

Innstillinger for webapplikasjonen er også ferdig. Her har gruppen lagt inn funksjonalitet for tempo, skriftstørrelse i tillegg til tekst- og bakgrunnsfarge.

8.1.3 Optimalisert for mobilbruk

Teamet har gjort webapplikasjonen svært mobilvennlig, da den nå forandrer seg ut ifra skjermens størrelse.

8.2 Videre arbeid

8.2.1 Mobilapplikasjon

En mobilapplikasjon for iOS og Android vil i fremtiden kunne implementeres, men per dags dato er mobilbruk gjennom nettleser første prioritet og vil jobbes mest med. Noen funksjonaliteter kan det være greit å lage egne spesifikasjoner for i forhold til hvilket operativsystem man er på. Som tilleggsfunksjon kan det også være lurt at man i fremtiden implementerer touch for scrolling.

8.2.2 Innlogging

Teamet ser for seg en innlogging som gir forskjellige brukere forskjellige brukertilganger og funksjonaliteter. Da kan for eksempel admin utføre endringer på selve innholdet i nyhetssendingen, mens ankeret bare har tilgang til det som er relevant for dem.

8.2.3 Lagring av innstillinger

Teamet ser for seg at man skal kunne lagre de forskjellige innstillingene lokalt på enheten sin, om det er en pc eller mobil. Dette vil gjøre prosessen mer sømløst da admin og anker slipper å endre innstillinger for hver gang man går ut og inn av applikasjonen. Denne type lagring krever ikke stor plass og teamet tror lokal lagring vil være ideelt for dette.

9 Referanser

Bakken, P, (2016). Lesehastighet og hurtiglesing. studenttorget.no.
Tilgjengelig fra:
https://studenttorget.no/index.php?show=41&expand=3795,41&artikkelid=6203
(Hentet: 4 mai 2021).

Docs.microsoft.com. (2021). A Tour of C# - C# Guide.
Tilgjengelig fra: https://docs.microsoft.com/en-us/dotnet/csharp/tour-of-csharp
(Hentet: 16 april 2021).

Dotnet.microsoft.com. (2021). Blazor | Build client web apps with C# | .NET.
Tilgjengelig fra: https://dotnet.microsoft.com/apps/aspnet/web-apps/blazor
(Hentet: 16 april 2021).

Git-scm.com. (2021). Git.
Tilgjengelig fra: https://git-scm.com/
(Hentet: 4 mai 2021).

Glasspaper. (2021). Hva er egentlig Scrum?
Tilgjengelig fra: https://www.glasspaper.no/artikkel/hva-er-egentlig-scrum
(Hentet: 16 april 2021).

Goswami, A., (2017). What, Why and How About SignalR.
Tilgjengelig fra: https://www.c-sharpcorner.com/UploadFile/abhijmk/what-why-and-how-about-signalr/
(Hentet: 1 mai 2021).

Liseter, I., (2020). scrolling – Store norske leksikon.
Tilgjengelig fra: https://snl.no/scrolling
(Hentet: 28 april 2021).

Netinbag.com. (2021). Hva er en teleprompter?.
Tilgjengelig fra: https://www.netinbag.com/no/technology/what-is-a-teleprompter.html
(Hentet: 23 april 2021).

Ntnu.no. (2021). Harvard-eksempler - Oppgaveskriving - NTNU.
Tilgjengelig fra: https://www.ntnu.no/viko/harvard-eksempler
(Hentet: 16 april 2021).

Persvold, A., (2019). administrator. Store norske leksikon.
Tilgjengelig fra: https://snl.no/administrator
(Hentet: 4 mai 2021].

Prosjektveiviseren.no. (2021). Prosjektstyring og smidig utviklingsmetodikk | Digitaliseringsdirektoratet.

Tilgjengelig fra:
https://www.prosjektveiviseren.no/prosjekttyper/digitaliseringsprosjekter/programvareutvikling/prosjektstyring-og-smidig-utviklingsmetodikk
(Hentet: 16 april 2021).

Reactjs.org. (2021). React.
Tilgjengelig fra: https://reactjs.org/
(Hentet: 10 mai 2021).

Rolstadås, A., (2021). Gantt-diagram. snl.no.
Tilgjengelig fra: https://snl.no/Gantt-diagram
(Hentet: 4 mai 2021).

Swagger.io. (2021). API Testing.
Tilgjengelig fra: https://swagger.io/solutions/api-testing
(Hentet: 16 april 2021).

Vizrt, (2021). About Vizrt.
Tilgjengelig fra: https://www.vizrt.com/vizrt
(Hentet: 15 mai 2021).

Vizrt, (2021). Vizrt. [Bilde].
Tilgjengelig fra: https://www.vizrt.com/-/media/Images/Vizrt-Internal/Leadership-1920x1080.ashx
(Hentet: 16 april 2021).

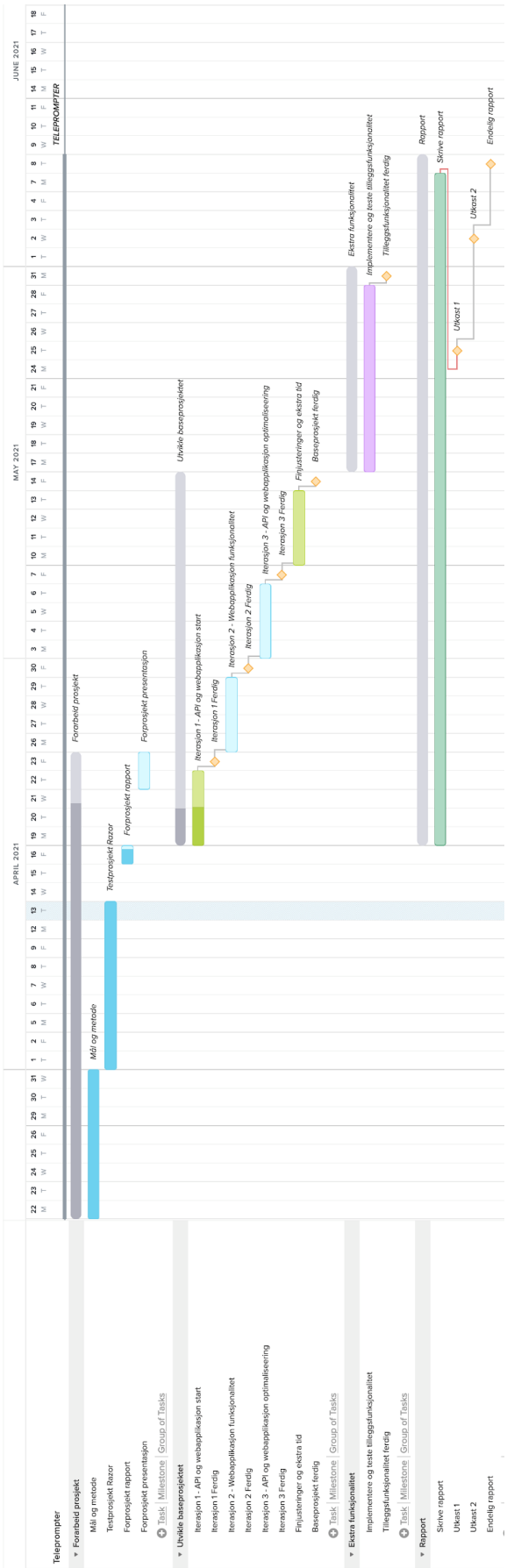
10 Vedlegg

10.1 Risikoliste

Sannsynlighet: Sannsynligheten for at risiko inntreffer.
Påvirkningsgrad: Påvirkningsgrad hvis risiko inntreffer.
Alvorlighet: Sannsynlighet multiplisert med påvirkningsgrad.
Risiko: 1-5 der 1 er lav sannsynlighet og liten påvirkningsgrad og 5 er stor sannsynlighet og stor påvirkningsgrad.

Risiko	Sannsynlighet	Påvirkningsgrad	Alvorlighet	Håndtering
Utilstrekkelig kunnskap om GitLab	4	2	8	Gruppen må lære seg GitLab skikkelig slik at man enklere kan samhandle med hverandre digitalt, med utgangspunkt i Covid-19.
Utilstrekkelig kunnskap om Vizrt sine systemer	3	5	15	Gruppen må jobbe tett med sine to veiledere i Vizrt for at det ikke skal oppstå misforståelser og forvirring.
Utilstrekkelig kunnskap om testverktøy	3	4	12	Gruppen må lese seg godt opp på bruk og anvendelse av testverktøy, slik at teamet får testet funksjonene i programmet på en god måte.
Utilstrekkelig kunnskap om Blazor	4	5	20	Gruppen har ikke vært borti Blazor tidligere, men med god hjelp av egne verktøy i tillegg til veilederne har teamet god tro på en bra løsning.

10.2 **GANTT-diagram**



10.3 **Ordliste**

Admin – «Innen IT er en admin en overordnet bruker i et datasystem» (Persvold, 2019, avsnitt 2).

Breakpoint – Et breakpoint er å sette et punkt i koden man ønsker å sjekke nærmere, hvor systemet stopper opp på dette punktet og avventer videre kjøring til når man selv ønsker.

C# - Programmeringsspråket teamet bruker til å utvikle applikasjonen.

GANTT-diagram – «Gantt-diagram er et diagram som viser aktiviteters tidsforløp i et prosjekt» (Rolstadås, 2021, avsnitt 1).

GIT – Er en gratis og åpen kildekode distribuert versjonskontrollsystem. (Git, 2021)

Item – Er innholdet i en Story. En story kan bestå av flere items. Henter ut Item fra story for å hente ut teksten som skal vises på skjermen.

React – Et JavaScript-rammeverk for bygging av UI. (React, 2021)

Readrate – Readrate eller lesehastighet er hvor fort du leser en tekst, målt i antall ord pr minutt. (Bakken, 2016)

Real-time app - Er en app som har en vedvarende samhandling mellom klient og server.

Scrolling – Scrolling er rulling av bilde eller tekst på en skjerm. (Liseter, 2020)

Story – En story er en del av en nyhetssending. Viz Mosart inneholder flere story objekter.

Teleprompter - En elektronisk enhet som viser tekst og signaler til en nyhetsanker. (Hva er en teleprompter?, 2021)

Viz Mosart – Et verdensledende studio-automasjonssystem. Programvare utviklet av Vizrt.