Høgskulen på Vestlandet

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

BACHELOROPPGAVE

Kryssplattform mobilutvikling med Flutter og Dart

Cross-platform mobile development with Flutter and Dart

Andreas Garvik

Morten Helland

Sondre Lindaas Gjesdal

Dataingeniør / IT

Fakultet for Ingeniør- og naturvitskap

02.06.20

Høgskulen på Vestlandet

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

TITTELSIDE FOR HOVEDPROSJEKT

Rapportens tittel: Kryssplattform mobilapplikasjon	*Dato:* 02.06.20
Forfatter(e): Andreas Garvik, Morten Helland , Sondre Lindaas Gjesdal	*Antall sider u/vedlegg:* 45 *Antall sider vedlegg:* 11
Studieretning: Dataingeniør / IT	*Antall disketter/CD-er:* 0
Kontaktperson ved studieretning: Bjarte Kileng	*Gradering:* Ingen
Merknader: Ingen	

Oppdragsgiver: Vizrt	*Oppdragsgivers referanse:* Ingen
Oppdragsgiver kontaktperson: Mikal H Henriksen	*Telefon:* 41687285

Sammendrag:

I denne oppgaven har vi laget en kryssplattform mobilapplikasjon for selskapet Vizrt. Mobilapplikasjonen skal snakke med Viz Story, som er en videoredigeringsløsning i form av en webapplikasjon. Den viktigste funksjonen er å kunne opplaste medieinnhold fra mobilen til Viz Story. Vi har laget kryssplattform mobilapplikasjonen i rammeverket Flutter med Dart som programmeringsspråk, begge utviklet av Google.

Stikkord:

Flutter	Viz Story	Mobilapplikasjon

Høgskulen på Vestlandet, Fakultet for ingeniør- og naturvitskap
Postadresse: Postboks 7030, 5020 BERGEN Besøksadresse: Inndalsveien 28, Bergen
Tlf. 55 58 75 00 Fax 55 58 77 90 E-post: post@hvl.no Hjemmeside: http://www.hvl.no

FORORD

Mobiltelefonen vår blir stadig en mer essensiell del av hverdagen vår. Den er lett å alltid ha med seg og rask å ta frem. Det fører til at mobilen gjerne er det første man trekker fram om man kommer over noe interessant. Vizrt ser at journalister ikke er et unntak og som leverandør av programvare ønsker å gjøre hverdagen mest mulig friksjonsfri. Vi har i denne oppgaven laget en løsning for Vizrt. Vi ønsker å takke Vizrt for denne muligheten og all hjelp vi har fått under utviklingen av applikasjonen. I denne oppgaven har vi utforsket, lært oss og fått gode kjennskaper til kryssplattform rammeverket Flutter, som er laget av Google. Flutter bruker og er delvis skrevet i Dart. Dart er et programmeringsspråk også utviklet av Google. Dart 1.0 ble lansert i oktober i 2011 og nåværende versjon er 2.8 per mai 2020. Flutter ble lansert i Alpha versjon i mai 2017 og kom ut i stabil versjon 1.0 i desember 2018. Den nåværende versjonen er 1.17 per mai 2020.

Link til blogg: https://bacheloroppgave-g18.firebaseapp.com/

TABLE OF CONTENT

1 INNLEDNING

1.1 Motivasjon og mål

Motivasjonen for prosjektet vårt kommer hovedsakelig fra arbeidsgiver. Vizrt ser at mobiltelefonen blir stadig mer brukt av journalister ute i felten til å ta bilder og video. Dagens opplastingsløsning går ut på at journalister som ønsker å laste opp medieinnhold til server må først laste innholdet fra mobiltelefonen til en datamaskin for så å kunne laste medieinnholdet videre opp til server. Vizrt har da tenkt at en mobilapplikasjon kan være en løsning som effektiviserer denne prosessen. Denne mobilapplikasjonen skal kunne direkte laste opp medieinnhold fra telefonen til server. En mobilapplikasjon vil så kunne være en mulighet for Vizrt å senere også kunne legge til mer funksjonalitet og videreutvikle programvaretilbudet deres til sine kunder.

Målet er da å utvikle en slik applikasjon for både iOS og Android. Vi skal ikke utføre dette med å lage to forskjellige applikasjoner, men å bruke en kodebase sammen med et kryssplattform rammeverk. Målet altså å få dette til å fungere med lik funksjonalitet på begge plattformer og få applikasjonen til å kjøre som en vanlig applikasjon på de respektive plattformene samt ha en Vizrt "look and feel". Rammeverket vil hjelpe oss stort i å kunne oppnå dette.

Vi har også egen motivasjon i dette prosjektet. Vi er motivert til å lære et nytt verktøy for utvikling av applikasjon på flere plattformer. Kryssplattform rammeverket vi skal bruke er ganske nytt og spennende og lover høy effektivitet og "programmeringsglede". Det er da interessant å prøve ut dette verktøyet for å se om vi klarer å lage en fullverdig og produksjonsklar applikasjon på både iOS og Android. Vi er videre motivert til å utføre et ordentlig prosjekt for en arbeidsgiver hvor det eksisterer et behov for en løsning. I løpet av bachelorutdanningen har vi laget flere prosjekter, men alle har fram til nå stort sett vært små, og gjerne ikke hatt noe bruksområdet i den virkelige verden. Vi ser nå frem til å kunne vise hva vi har lært i løpet av studiet og se om vi klarer å løse et etterspurt problem. Til slutt er vi også motivert fra genuin interesse for problemløsning. Det å løse et problem og faktisk få noe til å fungere er kanskje en av de beste følelsene man kan ha. Det er kjekt å oppnå noe, få tilbakemelding på godt arbeid og se at det man har bygget blir likt. Det er noe særegent med det man selv har laget og en glede i byggingen.

1.2 Kontekst

Vizrt er verdensledende leverandør av verktøy for visuell historieformidling innen broadcast, sport, e-sport og digitale medier. Vizrt tilbyr markedsdefinerende softwareløsninger for real-time 3D-grafikk, videoavspilling, studio-automasjon, sportsanalyse, media asset management og journalistverktøy. Et av disse journalistverktøyene heter Viz Story og er en videoredigeringsløsning. Den er laget for å kunne hente inn media, redigere, legge på grafikk og publisere video så raskt og effektivt som mulig på flest mulig kanaler. Viz Story brukes av journalister fra flere store mediehus rundt om i verden, og video publisert fra Viz Story blir vist i alt fra nyhetssendinger i beste sendetid til nettaviser, Twitter og Facebook-poster.

1.3 Avgrensninger

En avgrensning vi har i dette prosjektet er tid. Tidsberegning i et utviklingsprosjekt er alltid utfordrende. Det er flere uventede problemer vi kan støte på i løpet av utviklingsprosessen som viser seg for å være tidkrevende. Det fører til at kompleksiteten av applikasjonen som kan bli utviklet er begrenset. Andre avgrensninger i forhold til funksjonalitet finnes på serveren og programmet denne applikasjonen skal snakke med. Serveren har et avgrenset API som applikasjonen vi utvikler kan benytte seg av. Det blir ikke en del av oppgaven vår å gjøre noe endringer på noe programvare Vizrt allerede har, kun lage en ny app som skal fungere som et tillegg og skal benytte funksjoner som allerede finnes. Rammeverket Flutter som vi tar i bruk er under konstant utvikling, og siden det er ganske ferskt finnes mulig områder hvor det ikke skinner. Det er da heller ikke vår jobb i denne oppgaven å utvikle rammeverket til å fungere best mulig til Vizrt sitt behov, men å bruke det som allerede finnes av funksjonalitet og lage en løsning. Det betyr at eventuelle problemer vi støter på under utvikling av mobilapplikasjonen , det kan vært at i fra kompabilitetsproblemer til ytelses utfordringer, ikke blir blir vår jobb i fikse i denne oppgaven.

1.4 Ressurser

Til opplæring og for å bli flinkere i rammeverket, verktøyene og kunnskapen vi trenger for å gjennomføre dette prosjektet bruker vi flittig internett. Flutter sin dokumentasjon <https://flutter.dev> er veldig fullverdig og beskriver godt bruken av rammeverket. Den inneholder guider og eksempler for bruk av komponentene som kommer med rammeverket. Flutter og Dart er open source, noe som gjør at det finnes flere plugins <https://pub.dev> til rammeverket som også er open source, som vi kan bruke i vår applikasjon. Vi har også en ressurs i arbeidsgiver hvor vi har flere erfarne programmerere på teamet vårt. Disse har mulighet til å svare på spørsmål vi måtte ha både om deres applikasjoner og mobilutvikling generelt. Arbeidsgiver har også en del design komponenter som vi skal benytte oss av for å få applikasjonen til å ha deres designspråk. Programvare som VS Code <https://code.visualstudio.com> fra Microsoft og Adobe XD <https://www.adobe.com/no/products/xd.html> fra Adobe vil bidra til å gjøre utviklingsprosessen effektiv.

1.5 Oppbygging av rapporten

Kapittel 1 - Innledning og overbyggende generell informasjon om prosjektet og rapporten

Kapittel 2 - Prosjektbeskrivelse som inneholder informasjon om prosjektet og hvilken kunnskap vi har i starten av prosjektet.

Kapittel 3 - Design av prosjektet, her finner du forslag til løsninger i begynnelsen av prosjektet samt valg av løsning, verktøy og metodikk vi har brukt.

Kapittel 4 - Detaljert informasjon om design, hvordan vi har kommet fram til designet og hvorfor ting ser ut som det ser ut.

Kapittel 5 - Evaluering, hvordan vi har kommer til å evaluere resultatet av prosjektet og selve resultatet av våre evalueringer.

Kapittel 6 - Resultatene for vårt prosjekt, her kan du lære om testene og tilbakemeldinger vi har fått på mobilapplikasjonen.

Kapittel 7 - Diskusjon om våre opplevelser med å utvikle denne mobilapplikasjonen.

Kapittel 8 - Våre konklusjoner for dette prosjektet og hva som må bli jobbet videre med på mobilapplikasjonen

Kapittel 9 - Les gjennom referansene vi har brukt.

Kapittel 10 - Appendiks, se gjennom detaljert informasjon som ville vært for tung for hoveddelen av rapporten

2 PROSJEKTBESKRIVELSE

2.1 Praktisk bakgrunn

2.1.1 Prosjekteier

Prosjekteier er formelt Mikal H Henriksen ved Vizrt. Sammen med ham har vi et team fra Vizrt som gir oss veiledning og følger med på utviklingen. Dette teamet inkluderer personer med erfaring fra mobilutvikling og personer med erfaring fra grafisk design. Vizrt er leverandør av programvareløsninger til ulike mediehus. De er et norsk selskap med hovedkontor i Bergen Media City, med over 700 ansatte i 30 kontorer verden over med 52 nasjonaliteter. De leverer løsninger til blant annet CNN, CBS, NBC, Fox, BBC, BSkyB, Sky Sports, Al Jazeera, NDR, ZDF, Star TV, Network 18, Tencent og TV2.

Vizrt er et privateid selskap med Nordic Capital Fund VIII <https://www.nordiccapital.com/about/funds-vehicles/fund-vii/> som eier. Prosjektet kommer til å bli overtatt og videre vedlikeholdt av Vizrt etter vi er ferdige med oppgaven. Prosjektet er viktig for prosjekteier på bakgrunn av Vizrt er et selskap som leverer programvare til medieselskaper. De ønsker å gi kundene sine de beste, enkle og mest effektive løsningene og bidra til å gjøre arbeidshverdagen til journalister mest mulig friksjonsløs. Ved denne applikasjonen vil Vizrt kunne løse et stort effektivitets- og frihetsbehov for journalister. Journalister vil kunne laste opp bilder og video mens de er på reise, uten å måtte ha med seg en pc. De vil også kunne laste opp medieinnhold minuttet senere det har blitt tatt, som gjør at de kan dele medieinnhold til medarbeidere som gjerne sitter på kontoret og kan begynne å redigere medieinnholdet.

2.1.2 Tidligere arbeid

Det er ikke gjort noe direkte tidligere arbeid med et mobilopplastningsalternativ til Viz Story. Webapplikasjonen Viz Story finnes, men har ikke støtte for mobiltelefoner. Denne webapplikasjonen snakker med en server hvor alt medieinnhold som opplastes og finnes på Viz Story blir lagret. Protokollen som brukes i denne kommunikasjon består av HTTP og XML i et format kalt Atom. Se Appendiks 10.4 Atom Feed for mer informasjon. Denne protokollen har vi ikke noe med og det blir vår jobb å få den til å få applikasjonen vår til å snakke med serveren. Det er ikke en del av oppgaven å videreutvikle serveren eller gjøre noe endringer på den, så forhåpentligvis vil de kunne fungere sammen slik serveren er implementert i dag. Arbeidsoppdraget vårt handler om å lage en ny applikasjon. Applikasjonen vår må da kunne fungere med eksisterende APIer, servere og applikasjoner. Vi må da sette oss inn i en del kode og arkitektur slik at vi kan kunne integrere mobilapplikasjonen inn i eksisterende tilbud og løsninger Vizrt leverer. Dette handler blant annet om at mobilapplikasjonen må kunne bli tilpasset til det enkelte selskapet. Hver kunde av Vizrt sine løsninger kjører sin egen kopi av Vizrt sin programvare på egne servere. Dette er noe vi må ta hensyn til i applikasjonen. Vi må altså ta hensyn til at alle brukere i applikasjonen skal kunne velge sin egen server å laste opp medieinnholdet til.

Ellers tidligere arbeids omfatter Vizrt sitt designspråk. Vizrt har sine egne design komponenter i programmet Adobe XD som er ønsket skal også brukes i vår applikasjon. I tillegg har de et eget bibliotek av ikoner. Det blir da en utfordring å få oversatt disse komponentene til et nytt språk, men Adobe XD har verktøy som kan kan hjelpe oss i det arbeidet. Sammen med rammeverket Flutter kommer det også en del ferdige komponenter laget i Material Design <https://material.io/design> retningslinjene. Material Design er et designspråk som Google har utviklet som kom ut i 2014. Det blir da en utfordring å få dette integrert sammen.

2.1.3 Initielle krav

De initielle kravene til mobilapplikasjonen var:

- Kryssplattform applikasjon
- Opplasting av filer
- Bakgrunnsopplastning
- Mulighet å ta bilder og video i applikasjonen
- Intro for førstegangsbruk av applikasjonen

Det viktigste kravet var at applikasjonen skulle kunne virke på både iOS og Android. Mer detaljert om brukeropplevelse var kravene til denne applikasjonen:

- En bruker skal kunne skrive inn en URL som skal linke applikasjonen til den rette serveren for sitt selskap
- En bruker skal så kunne se gjennom medieinnholdet på telefonen og velge ønsket media.
- En bruker skal kunne se over medieinnholdet som er lagt til i applikasjonen i en liste og kunne slette uønskede filer som ble lagt til, for eksempel med et uhell, samtidig så de andre filene blir bevart. Filene skal presenteres i en slags kø, som enkelt viser rekkefølgen på opplastingen.
- En bruker skal få tilbakemeldinger fra applikasjonen om opplastingsprogresjonen, hvor langt en fil er kommet i å bli lastet opp, i form av en progresjonsindikator, hvilken fil som er den som for øyeblikket blir lastet opp og hva som er rekkefølgen på hva som skal til å bli lastet opp.
- En bruker skal så ha mulighet til å kunne endre på rekkefølgen filene blir lastet opp i både før opplastingen har begynt og mens filer blir lastet opp.
- En bruker skal kunne stoppe hele opplastingen av filer og kunne kun fjerne enkelte filer fra opplastingskøen. Om filen som holder på å bli lastet opp fjernes fra køen skal opplastingen av filen også stanses.
- En bruker skal kunne få beskjed om opplastingen var en suksess eller vise eventuelle feil som oppstår ved opplastning av de enkelte filene.
- En bruker skal motta varsler på mobiltelefonen om opplastingen ved å vise antall filer som har blitt lastet opp og hvor mange som er igjen. Det skal også vises varsel om eventuelle feil under opplastning eller at opplastingen er ferdig.

Videre skal applikasjonen også kunne motta innhold delt fra en annen applikasjon via mobiltelefonens delingsgrensesnitt, avhengig av de ulike løsningene tilbudt av operativsystemet. Applikasjonen skal også følge Vizrt sitt designspråk.

Videre skal applikasjonen kunne lastes ned og kjøre på både iOS og Android. Etter at denne kjernefunksjonaliteten var på plass, ønsket Vizrt at mer funksjonalitet skulle kunne bli lagt til applikasjonen. Dette var blant annet funksjonalitet og oppkobling mot Vizrt One som er et media asset management (MAM) system <https://www.vizrt.com/digital/asset-management>. Et media asset management (MAM) system er i Vizrt sin sammenheng et system som gjør det enkelt å holde styr på mediefiler for de forskjellige departementene og gjør det mulig for et selskap å dele og samarbeide sammen med medieinnholdet på tvers av selskapet. Det er et verktøy hvor man kan søke, redigere og publisere media.

2.1.4 Initiell løsnings-idé

Den initielle løsningen, foreslått av Vizrt, for å laste opp til Viz Story direkte fra mobil blir å lage en mobilapplikasjon. De hadde fått forespørsler fra kundene sine at brukere av Viz Story ønsket en mindre tungvint måte å opplastet medieinnhold fra mobiltelefonen. Dagens tungvinte løsning er altså at man må først koble mobiltelefonen til en datamaskin, så laste medieinnholdet fra mobiltelefonen til datamaskinen, for så å kunne laste opp medieinnholdet til VizStory. Vizrt har da tenkt at en mobilapplikasjon kan være løsningen på å gjøre denne prosessen enklere og mindre tungvint. Det å lage en separat mobilapplikasjon ville også gjøre det mulig å legge til flere funksjoner etterhvert. Vizrt har ikke mye erfaring med å lage mobilapplikasjoner, noe som fører til at de ikke har mye innarbeidede metoder for utvikling. Heller ikke med noen tidligere prosjekter som omhandler mobilapplikasjons-utvikling. Derfor hadde ikke Vizrt noen sterk preferanse på hvordan vi skulle utvikle mobilapplikasjonen. Vizrt lot det så være opptil oss å velge rammeverk, språk og verktøy. Noe som førte til at vi hadde stor frihet til å velge det vi hadde lyst til selv. Det at de heller ikke hadde noe sterkt ønske om programmeringsspråk eller rammeverk kan også by på utfordringer da vi står mer på egne ben, og ikke kan utnytte kunnskap Vizrt hadde hatt i teamene sine. Vi endte opp med å velge Flutter, som er et rammeverk for utvikling av applikasjoner på flere plattformer, laget av Google. Den største fordelen med rammeverket er at man kun har en kodebase og kan kompilere koden til å kjøre som maskinkode på de ulike plattformene <https://flutter.dev/docs/testing/build-modes>.

2.2 Litteratur om problemstillingen

Informasjonen om prosjekteier, Vizrt, som omhandler hva de er og hva de gjør er hentet fra deres egne nettsider på About Vizrt <https://www.vizrt.com/vizrt>. Videre er informasjonen om deres videoredigeringsløsning Viz Story også hentet fra deres egne nettsider på Viz Stoy <https://www.vizrt.com/products/viz-story>. Informasjon om rammeverket Flutter er hentet fra <https://flutter.dev/docs>. Rammeverket er open source og man har full oversikt over hva som foregår samtidig som det finnes godt med dokumentasjon på nettsiden. Informasjon om Atom protokollen stammer fra <http://www.atomenabled.org> og IETF <https://www.ietf.org/>. Videre er Material Design hentet fra <https://material.io>.

3 DESIGN AV PROSJEKTET

3.1 Forslag til løsning

3.1.1 Tilpasning av webapplikasjon

Viz Story er en webapplikasjon, noe som i teorien gjør at den er tilgjengelig på mobil, men det er ikke gjort noen tilpasninger for at det skal være en god løsning. Så det hindrer at den kan brukes på mobil. Webapplikasjonen måtte da ha blitt tilpasset responsivt slik at den hadde vært brukbar på mobil. Løsningen innebærer at vi må lære oss programmeringsspråket og verktøyene som er brukt i webapplikasjonen og at vi må bruke tid på å sette oss inn i en eksisterende kodebase. Uten at Vizrt foreslo denne løsningen er det i teorien mulig å gjennomføre.

3.1.2 Kryssplattform mobilapplikasjon

Denne løsningen innebærer å lage en felles applikasjon som fungere på tvers av plattformer. Denne applikasjonen skal snakke med samme server som Viz Story og gjøre det mulig å laste opp forskjellig medieinnhold fra telefonen. Mobilapplikasjonen skal kunne kjøre på både iOS og Android med samme kodebase. Løsningen innebærer at vi må lære oss et kryssplattform rammeverk og kunnskap om generell mobilapplikasjons-utvikling er et krav.

3.1.3 Ulike applikasjoner til den enkelte plattformen

Denne løsningen innebærer at man lager to forskjellige applikasjoner, en til iOS i programmeringsspråket Swift <https://developer.apple.com/swift/> eller eldre Objective-C og en til Android i programmeringsspråket Kotlin <https://developer.android.com/kotlin> eller eldre Java. Grunnen til dette er at utvikling av mobilapplikasjoner på de forskjellige plattformene låser deg til hvilke programmeringsspråk som kan brukes. Begge applikasjonene skal ha samme funksjonalitet og snakke med samme server som Viz Story og gjøre det mulig å laste opp forskjellig medieinnhold fra telefonen. Applikasjonene skal ha likt design, oppføre seg likt og ha samme brukeropplevelse. Løsningen innebærer at vi må lære oss to ulike programmeringsspråk og to forskjellige plattformer. Vi må også lære oss to forskjellige domener i mobilapplikasjons-utvikling.

3.1.4 Diskusjon av alternativene

Å skrive om webapplikasjonen er mulig løsning, da det kunne vært flere fordeler. En av fordelene hadde vært at det videre i fremtiden da kun hadde vært en applikasjon å gjøre endringer på eller legge til ny funksjonalitet som for eksempel å støtte andre medieformater. Videre kunne løsningen blitt mer brukervennlig da brukeren kun hadde hatt en applikasjon å lære seg, bruke og bli kjent med. Det hadde da altså ikke vært noen stor læringskurve for brukere som allerede bruker webapplikasjonen å få lastet opp innhold fra telefonen. Ulempen med en slik løsning hadde vært at applikasjonen da måtte ha gjennomgått en stor endring. Det måtte ha blitt avgjort hvilke funksjoner som skulle vært tilgjengelige på mobilversjonen. Webapplikasjonen måtte også da ha blir responsive i forhold til ulike skjermstørrelser og om all funksjonalitet skulle fremdeles vært i webapplikasjonen hadde det mulig ført til et par begrensninger på applikasjonen. Det kunne ført til at store deler programmeringskode måtte ha blitt skrevet om. Det i seg selv hadde vært en stor utfordring da vi hadde hatt en stor kodebase vi måtte blitt kjent med før vi kunne ha gjort endringer.

Løsningen om å lage en selvstendig mobilapplikasjon enten kryssplattform eller to ulike til hvert operativsystem har både fordelen og ulempen ved å begynne helt fra start. En fordel er at man ikke trenger å sette seg inn i tidligere skrevet kode og brukt arkitektur. Man vil ha enkelt kunne ha et forhold til koden i applikasjonen siden all kode er skrevet av oss. Videre er fordelen at vi kan ta i bruk nyere og mer moderne teknologier. Vi står mer fri til å velge verktøy og programmeringsspråk. Det vil potensielt gjøre utviklingsopplevelsen og effektiviteten vår bedre. Ulempen er at det vil være mye oppsett kode som må skrives bare for å få en minimal applikasjon til å virke. Vi må også bestemme oss for arkitektur, noe som er kostbart om vi velger feil. Vizrt bestemte sammen med oss at vi skulle lage en mobilapplikasjon, noe vi var motivert til å utføre.

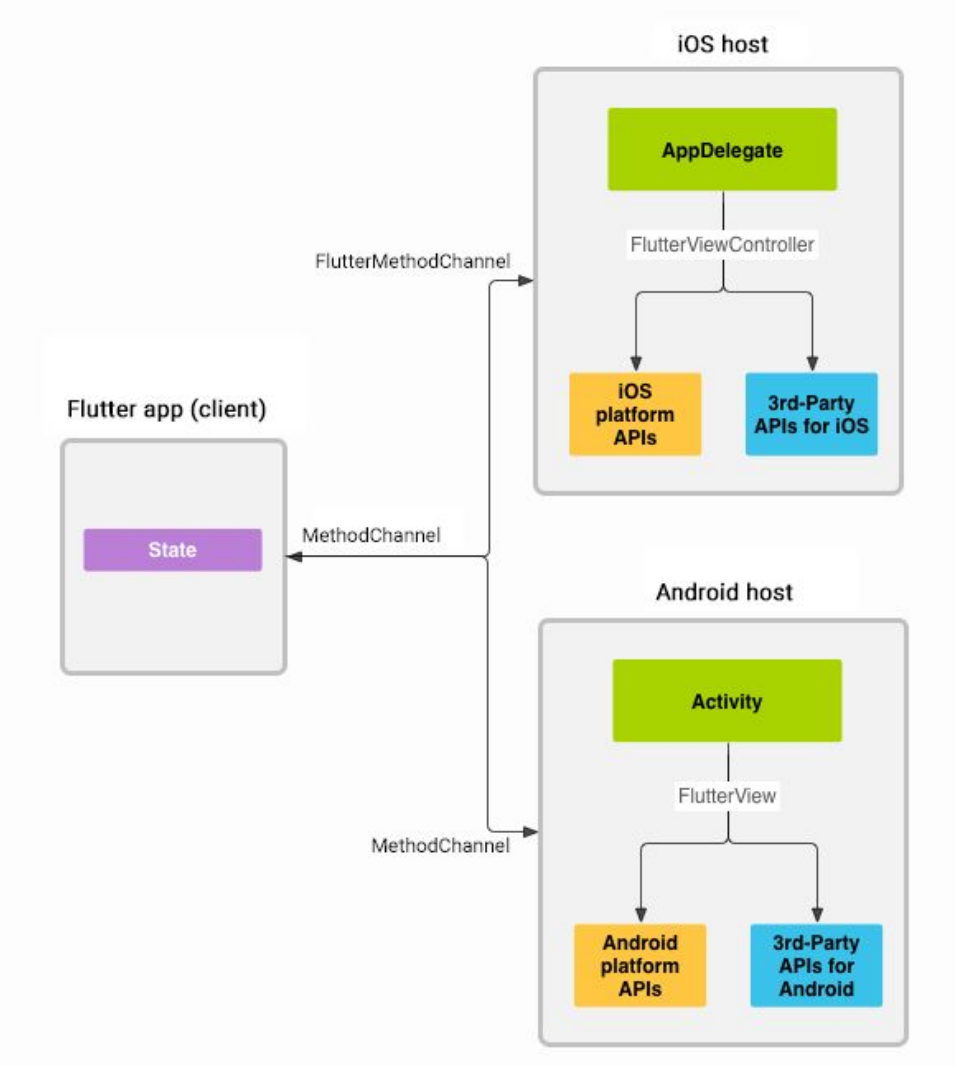
Da sto valget mellom å lage en kryssplattform applikasjon eller å lage to mobilapplikasjoner, en til hver plattform. Fordelen med å velge kryssplattform applikasjon fremfor en til hver plattform er at vi kan ha en samlet kodebase hvor vi kan ha all applikasjonslogikk. Denne vil bli delt mellom de ulike plattformene. Alternativet er at vi må skrive de samme funksjonene to ganger i to forskjellige språk. Her kommer vi innpå et viktig punkt. Ved å velge kryssplattform applikasjon trenger vi bare å lære oss et programmeringsspråk og vi kan da ha tid til å lære oss det godt. Vi får også mer tid til å optimalisere koden og funksjonene i dette ene språket. Alternativet her igjen er å måtte lære oss to forskjellige språk og optimalisere koden og funksjonene på to språk. Senere og når vedlikeholding trengs vil en kryssplattform applikasjon ha fordelen med kun en kodebase. Det vil bety halvparten av problemer og bugs og halvparten av tiden for å vedlikeholde applikasjonen. Ulempene med en kryssplattform applikasjon er at det kan være problemer med kompatibilitet av ulik plattform spesifikk funksjonalitet. Man kan risikere at det kan ta litt tid fra en ny funksjon er lansert på en plattform at en pakke eller løsning som kobler den funksjonaliteten til kryssplattform rammeverket finnes. Ta dette eksempelet som inkluderer Apple sitt programmeringsspråk Swift for iPhone og Flutter som er et kryssplattform rammeverk laget av Flutter:

Apple kan lansere en ny funksjon på iPhone som de gjør tilgjengelig i Swift, men de vil ikke lage løsningen tilgjengelig i Flutter. Da må enten Flutter teamet eller noen andre utviklere lage en kobling mellom funksjonaliteten og rammeverket slik at den kan brukes i Flutter. Dette kan vært alt i fra nye sensorer til endringer i applikasjonens livssyklus.

I Flutter gjøres dette ved å gjøre det mulig å skrive plattform spesifikk kode og kalle den via et grensesnitt over en plattform-kanal, hvor meldinger blir sendt mellom en klient, som er brukergrensesnittet, og en vert, som er plattformen, bedre forklart med dette diagrammet hentet fra Flutter dokumentasjonen

<https://flutter.dev/docs/development/platform-integration/platform-channels>,

se neste side:



Til venstre er en kjørende Flutter applikasjon, som har sin egen tilstand. Med tilstand menes data som finnes i applikasjonen under kjøring. I vårt eksempel kan dette vært filer brukeren har lagt til i applikasjonen. I tillegg avhengig av hvilken plattform applikasjonen kjører på så har applikasjonen en kjørende prosess i operativsystemet på mobiltelefonen, altså en "Activity" som det heter på Android eller en "AppDelegate" som det kalles på iOS. Flutter applikasjonen kan kommunisere med denne prosessen via en kanal. Så når Flutter applikasjonen ønsker å utføre en funksjonalitet på vegne av brukeren som for eksempel klikket på en knapp i applikasjonen, sender Flutter applikasjonen en melding på kanalen med en forespørsel om at prosessen kan utføre dette og gi resultatet tilbake til Flutter applikasjonen. Dette betyr at Flutter kjøres omgivende en Android eller iOS applikasjon.

Fordelen med måten Flutter gjør dette på er at Flutter kan da fremdeles, selv om det er et kryssplattform rammeverk, enkelt holde følge med evolusjonen til de ulike plattformene.

Fordelene med å velge å lage to applikasjoner, en til hver plattform er først og fremst å ha det beste biblioteket med alle mulige funksjoner allerede bygget inn. Det innebærer at man vil kunne benytte hele plattformen og kunne bruke den akkurat som man vil. Det betyr at alt er støttet og programmeringsspråket og bibliotekene blir vedlikeholdt og oppdatert med en gang en ny funksjon ankommer. Det kan også fort være ytelse fordeler med det alternativet da noen kryssplattform applikasjoner kan ha flaskehalser når man skal oversette koden man har skrevet til kode som kan kjøres på plattformen. Det er spesielt løsninger som bruker en "bridge" <https://reactnative.dev/docs/native-modules-ios> som opplever slike problemer En "bridge" er et mellomledd som gjør det i React Native sitt tilfelle det mulig for applikasjonen å kommunisere med plattformen. Altså å kjøre Javascript på mobiltelefoner, kort fortalt ved at en Javascript prosesstråd kommuniserer med en plattform prosesstråd.

Til alternativet med kryssplattform-applikasjon står vi mellom et valg av rammeverk. Av rammeverk finnes det flere alternativer blant annet React Native <https://reactnative.dev>, Xamarin <https://dotnet.microsoft.com/apps/xamarin>, Ionic <https://ionicframework.com> og Flutter. Flutter er den nyeste av disse alternativene, noe som ikke nødvendigvis er det sikreste valget, men Flutter har en del fordeler som skiller det fra de andre. Hovedforskjellen er at Flutter ikke kjører på "bridge" som kobler koden til de ekvivalente kallene på den spesifikke plattformen, men kan kompileres AOT (ahead of time) til maskinkode som så kan kjøre på de spesifikke plattformen. Det er ved hjelp av programmeringsspråket Dart at Flutter har denne muligheten. Dart har også andre fordeler som øker utviklingsopplevelsen og effektiviteten, som "Hot Reload" <https://flutter.dev/docs/development/tools/hot-reload>. Med utviklingsopplevelsen menes hvor enkelt man kan skrive kode og implementere ønsket funksjonalitet, og i hvor stor grad man må kjempe imot rammeverket for å få til kjørende og fungerende produkt. Eksempler som gjør utviklingsopplevelsen bra i Dart er for eksempel "Hot Reload" verktøyet. Dette verktøyet gjør det mulig å endre på applikasjonens design eller grensesnittet uten å måtte kompilere koden eller starte applikasjonen for å se resultatet. Den største begrensingen med Flutter er at det er relativt nytt og ikke har så veldig mange eksterne biblioteker for all funksjonalitet enda.

3.2 Valgt løsning

Vizrt hadde egentlig bestemt seg for at det var en mobilapplikasjonen de ønsket å få laget. Det var derfor egentlig ingen særlig diskusjon vi var med på og vi hadde altså lite med hvordan avgjørelsen ble tatt. Om Vizrt intern hadde hatt en diskusjon på alternativene før de gikk ut med oppgaven ga de ikke noe inntrykk om, men vi var ikke noe uenige om den avgjørelsen. Videre etter å ha evaluert de ulike alternativene kom vi raskt frem til at vi ønsket å lage en kryssplattform applikasjon. Hovedårsaken til det valget var at vi er en gruppe på tre personer og det hadde vært en for stor oppgave i forhold til tidsrammen å både lære seg to språk og lage to applikasjoner. Vi ønsket å bruke tiden vi hadde på å heller lage en god og forhåpentligvis brukbar applikasjon fremfor to halvveis og ubrukelige applikasjoner. Så valget falt altså på å lage en kryssplattform applikasjon.

3.3 Valg av verktøy

Når det kom til valg av verktøy og i vårt tilfelle valg av kryssplattform rammeverk, var vi i grunnen ikke i tvil. Vi ønsket å bruke Google sitt rammeverk Flutter. Grunnen til at vi valgte å gå for Flutter først og fremst at Flutter kan kompileres til maskinkode som kan kjøres på den enkelte enheten, dette er viktig for å kunne så den beste applikasjonen ytelsesmessig. Andre grunner er de gode utviklingsverktøyene Flutter har som "Hot Reload" og måter å visualisere det grafiske brukergrensesnittet.

Videre har valgt å adoptere strategien til Vizrt ved å bruke SCRUM sammen med Kanban, med sprinter som varer i en uke av gangen. For å kunne utvikle en app trenger vi noe å teste det vi skriver. Vi har valgt å bruke en Android emulator som ligger innebygd i Android Studio eller en iOS emulator som finnes på macOS og Xcode, i tillegg til egen mobil til å teste applikasjonen. Av utviklingsmiljø har vi valgt Visual Studio Code for sin enkelhet, men med fortsatt store muligheter for å tilpasse og gi støtte for rammeverket vi har valgt for å lage mobilappen.

3.4 Prosjektmetodikk

3.4.1 Utviklingsmetodikk

Vi har som tidligere nevnt arvet utviklingsmetodikken til Vizrt som er den agile utviklingsmetodikken SCRUM sammen med Kanban og sprinter som varer 1 uke. Vi har for oversikten sin skyld tatt i bruk Jira som er brukt til prosjektstyring og organisering av deloppgavene våre, og for bedre oversikt på hvor langt vi er kommet for Vizrt.

Vi har faste sprintmøter hver uke som følge av SCRUM. På disse møtene rapporterer vi hvor langt vi er kommet og hva vi har fått gjort den uken som har gått. Deretter går vi over på hva som skal skje den neste uken. Vi ble også enig etter det 3. møte at vi skal snakkes oftere med bare vår kontaktperson fra Vizrt etter forprosjektrapport er blitt levert. Etter påsken har vi hatt oftere møter med design ansvarlig og backend ansvarlig hos Vizrt etter det vi har trengt av støtte.

Møtene vi har hatt vart rundt 1 time og blitt brukt til å planlegge den neste uken og informere oss studenter om hva vi har i vente litt lengre framover i tid ettersom at vi ikke har vært med på et prosjekt av denne skalaen før. Vi har også fått tilbakemelding på arbeidet vi har gjort den forrige uken og informert om andre ting som er blitt arbeidet med og skal bli fokusert på for den neste uken.

3.4.2 Prosjektplan

Vi har som mål med dette prosjektet å lage en mobilapplikasjon til å laste opp bilder og filmer til Viz Story. Den nåværende løsningen er å laste opp film og bilde til sin egen datamaskin før man overfører mediene fra datamaskin til Viz Story. Dette er noe Vizrt er usikker på om det kommer til å bli brukt og ønsket derfor å lage en bacheloroppgave for å lage denne mobilapplikasjonen som de kan bruke til å teste på sine kunder.

Det vi ønsker å få i mål er en mobilapplikasjon som nevnt tidligere. Ettersom dette kom i havn etter rundt 2 uker med jobbing er vi i gang med ekstra arbeidsoppgaver som de har hatt på ønskeliste for en slik applikasjon. På ønskelisten ligger blant annet QR skanning for å få tak i URL til et Viz Story instans og køsystem for hvilke filer som skal bli lastet opp i hvilken rekkefølge. Her kommer det nok til å skje store endringer etterhvert som mobilapplikasjonen tar form. Vårt arbeidsomfang har vært veldig overkommelig ettersom de fleste kravene vi fikk etter uke 2 var for det meste mindre. For de tilleggsmål som er satt for oss er det ingen som er uoppnåelig innenfor den tid vi har.

Av viktige milepæler har vi følgende:

- Project kickoff - 9. Mars
- Kickoff meeting with firm - 16. Mars
- Meeting with supervisor - 18. Mars
- Meeting with firm - 23. Mars
- Meeting with firm - 30. Mars
- Pre Project report - 3. April
- Mobile App prototype - 6. April
- Status report - 28. April
- Temporary 1.0 Release - 30. April
- Draft 1(final report) - 19. May
- Draft 2(final report) - 26. May
- Temporary 1.1 Release - 29. May
- Final report - 26. May
- Reflection form - 10. June
- HVL EXPO - 11. June

Oppdragsgiver Mikal H. Henriksen har ikke kommet med noen spesifikk dato når prosjektet skal være ferdig. Vizrt som oppdragsgiver har heller ikke lagt fram noen krav for når mobilapplikasjonen skal være ferdig. Vi har fått beskjed om at vi kommer så langt som mulig og når det blir tid for å fokusere hovedsakelig på bachelor rapporten så skal vi gjøre det.

Kommunikasjon mellom gruppen og oppdragsgiver er veldig bra. Sett bort fra ukentlige sprintmøter er det ikke noen andre krav om rapportering. Til tider gjennom prosjektarbeidet har det kommet ønsker om å se deler av hva som er blitt gjort. Dette har fort blitt levert til de gjennom Microsoft Teams.

3.4.3 Risikovurdering

Begynnelsen av prosjektet vårt er blitt kraftig preget av COVID-19 pandemien og den midlertidige stengningen av Vizrt sine kontorer. Dette har ført til nye utfordringer som vi ikke hadde sett for oss når det kom til kommunikasjon og arbeidsflyt. Vi føler dette er blitt løst på den beste måten mulig situasjonen tatt i betraktning. Det vi oppdaget i de første par ukene er at til tross for god kommunikasjon, åpner den elektroniske kommunikasjonsplattformen for at det lettere oppstår mindre misforståelser. I starten av prosjektet gikk dette heldigvis bare utover hva som var planen framover og ikke noe arbeid som faktisk var gjort. I forhold til endringer av kravspesifikasjoner føler vi oss ganske så trygg ettersom at Vizrt sitt Viz Story team er godt erfarne utviklere og har med dette et realistisk syn på hva som er oppnåelig innenfor det tidsperspektivet vi har. I halvdel av prosjektet har vi jobbet videre som første halvdel. Ting har normalisert seg rundt omstendighetene med å måtte jobbe hjemme i siste halvdelen av prosjektet. Det har heldigvis ikke oppstått noen problemer med dette så her har vi vært veldig heldig. Ettersom det er stor fallhøyde her med å få til over nett. Heldigvis har IDE-en vi bruker tilgang til noe som heter liveshare, der to stykker kan sitte å utvikle i lag, dette hjelper veldig og er en grei erstatning til å sitte ved samme datamaskin. Alt i alt har vi endt opp med et bra produkt og unngått flere ting som kunne gått galt. Se vår risikoliste i kap 10.1.

3.5 Evalueringsplan

Vi evaluerer prosjektet vårt på litt forskjellige måter. Første prioritet vil være at all kode fungerer som ønsket. Vi har noen unit tester for dette og planlegger å utvide antall unit tester parallelt med skriving av denne rapporten. For å gjøre jobben vår enklere har vi også tatt i bruk en populær CI(Continuous Integration) som heter Travis CI <https://travis-ci.org> for å gjøre det lettere å sjekke om unit testene er vellykket.

Vi har gjort flere tester på integrasjon for å sjekke at modulene vi har skrevet henger sammen. Dersom modulene våre ikke fungerer sammen får vi store problemer med å sende rett filinformasjon opp til serveren eller i det hele tatt å koble seg til serveren i første omgang. Det skal da heller ikke mye til for at hele mobilapplikasjonen krasjer. Det vil også skape store problemer for brukeren på sin ende, samt skape problemer med tilkobling til server.

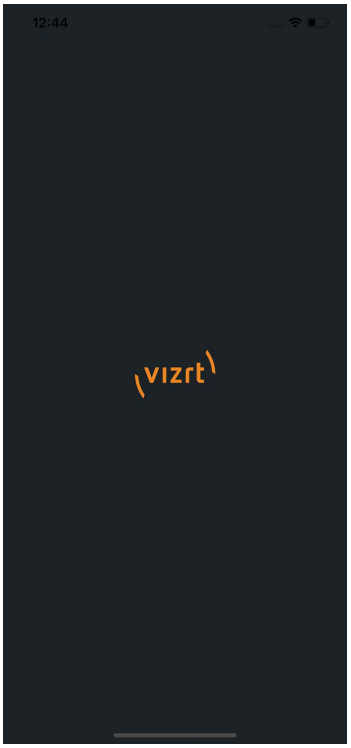
Vi har fått en journalist fra TV2 til å teste vårt produkt og venter på øyeblikket på svar derfra. Journalisten vil få tilgang til en iOS beta versjon av appen. Ettersom at dette er en person som er i målgruppen for appen vil denne tilbakemeldingen ha mye å si for hvordan vi evaluerer hvor bra brukeropplevelse vi har fått til. Samtidig får vi tilbakemelding på brukeropplevelsen fra flere personer fra Viz Story sitt team. Ettersom vi ikke har mye erfaring innenfor dette emnet i en profesjonell setting er dette satt mye pris på.

Vi har siden begynnelsen hatt veldig stort fokus på brukerkravene og få disse godkjent. Disse var som tidligere nevnt ikke stort problem å få til. Vi har så langt fått tilbakemelding fra Viz Story team på at det tilfredsstiller deres brukerkrav og vi venter på tilbakemelding fra brukertesting om dette er bra nok til at de vil ønske å ta det i bruk.

4 DETALJERT DESIGN

Design er det første du tenker over når du ser et produkt. Du tenker ikke over hva funksjoner appen har eller hva som skal til for og utføre de. En vurdering av appen blir skapt i løp av noen få sekunder. Det er derfor første inntrykket av en app har stor betydning.

Et bra førsteinntrykk skaper vi med noe som heter splash screen, som gjør at den ventetiden mellom appen starter til appen er oppe og går, blir fylt med et grafisk bilde. Dette gjør at appen blir mer elegant og flyten i appen blir bedre. Man tenker ikke alltid over det, for i noen apper så går det så raskt å starte opp appen at du ikke for det med seg.



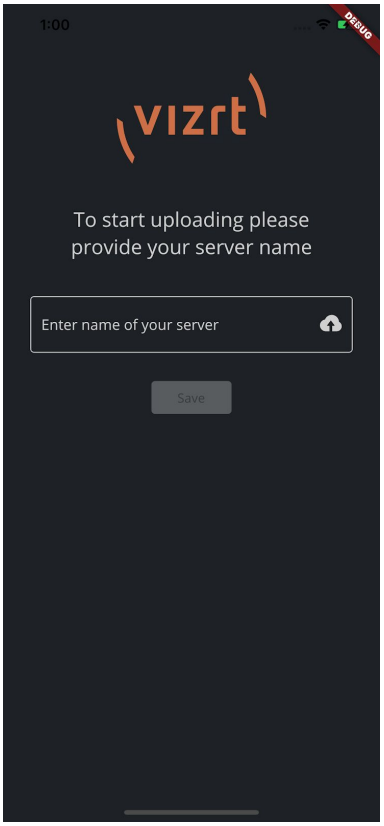
Figur 4.1: Splashscreen

Det som blir vist ovenfor er en splash screen(figur 4.1), som vi valgte og bruke. Derfor er det viktig at det ser bra ut, siden dette er det første man ser. Vi har valgt og bruke svart bakgrunn med oransje logoen, slik at det er ingenting som tar fokuset vekk i fra logen. For slik som det er no så ser man logoen og skaper en forventning over hva som skal skje.

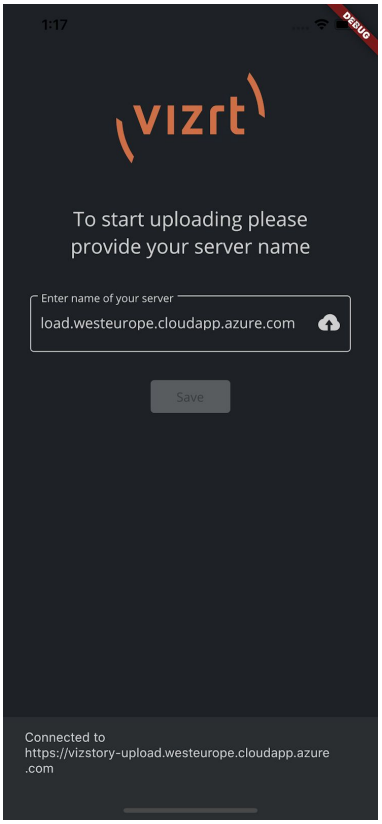
```dart
class Vizrt {
  static Color get vizrtGrey95 => Color(0xFF1E2226);
  static Color get vizrtGrey090 => Color(0xFF2A2E31);
  static Color get vizrtGrey85 => Color(0xFF36393D);
  static Color get vizrtGrey035 => Color(0xFFACAEAF);

  static Color get buttonDisabled => Color(0xFF595C5F);
  static Color get iconsTextButtons => Color(0xFFD0D0D1);

  static Color get success => Color(0xFF4DE18A);
  static Color get warning => Color(0xFFFCDC5C);
  static Color get fail => Color(0xA0F75B5E);

  static Color get darkOrange => Color(0xFFCF6F47);
  static Color get black => Color(0xFF000000);
  static Color get white => Color(0xFFFFFFFF);
  static Color get blueSelection => Color(0xFFA6078EA);
  static Color get blueHover => Color(0xFF4F6Be8);
```

Figur 4.2: fargekode.

Vi fikk tilgang til Vizrt sitt design på forhånd som vi kunne plukker element som vi ville. De hadde ingen krav på forhånd og vi fikk stort spillerom. Men vi brukte likevel mange av elementene. Som dere ser så har vi implementert alle fargekodene inn i appen, som gjør det enkelt å bruke de riktig fargekodene til riktig tid.



Figur 4.3: Welcome screen



Figur 4.4: Welcome screen info

Her ser man bilde over den første siden man kommer til i appen. Den ble lagt fordi man er nødt til å ha en server å laste opp til for at appen skal fungere slik som man vil. Det er flere ting og ta opp her angående designet som man kanskje kan stille spørsmål med. Hvorfor er input tekst såpass høyt oppe? hvorfor akkurat de fargene og knappene?

Plassering av input tekst:

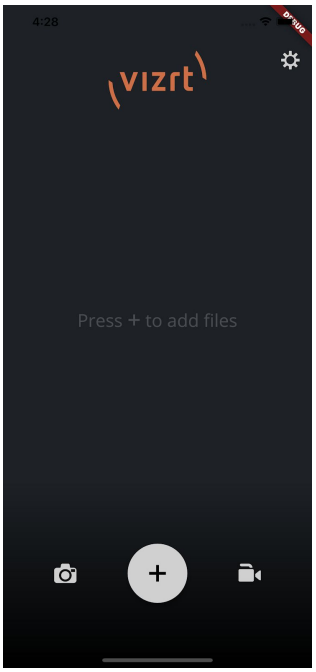
- Grunnen for den er såpass høyt er på grunn av 2 ting:
    - Hvis man har den midt på skjermen og logoen er på topp, så vil det ser ut som du får en varsling i steden for input. På en Iphone vil varsling og melding komme midt på skjermen. Dette kan føre en misforståelse hos brukeren
    - Det andre er at slik som alt er plassert no så blir likt alle type modeller, om det er en liten eller stor mobil.

Design:

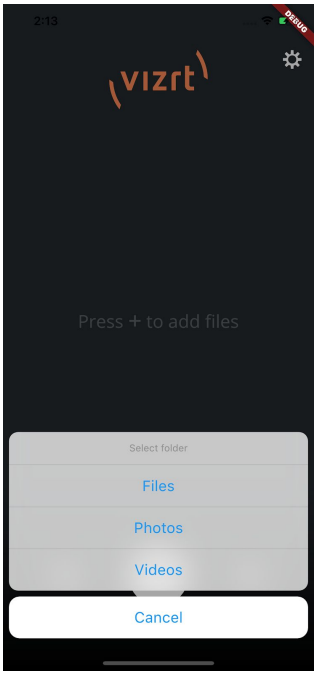
- Knapper ble designet på grunnlag av det som Vizrt bruker på sine plattformer.
- Skrifttype gjennom appen er Open Sans Regular
- Fargen på iconer og skrift er lysegrå som også er lik gjennom appen
- Icon ble valgt ut i frå Vizrt sitt design

På figur 4.4 så ser man nede på skjermen at det har dukket opp at du connected og vil bli overført til hovedsiden. Men det er en grunn for hvorfor du får varsling nede og ikke midt på skjermen. For alt handler om at du skal kunne kjenne deg igjen i appen, slik at ting som skjer i appen ville gjort på samme måte som en native app. Så derfor får man varsling nede i applikasjonen.

Videre så skal vi se på hvilken funksjoner denne applikasjonen kan tilby. For oppgaven vi fikk av Vizrt var at vi skulle laste opp bilder, video og lydfiler. Så videre skal vi ser hvordan design messig har løst denne oppgaven. Der vi skal gå gjennom hvordan alt er tilpasset slik at du skal kunne kjenne deg igjen og hvordan ting er plassert slik de er.



Figur 4.5: Main screen



Figur 4.6: Add screen

Her kan man se hovedsiden i appen, som viser funksjonene denne appen leverer. Alt på denne siden er nøye planlagt, så derfor skal vi gå gjennom denne siden i detalj. Bilde du ser til høyre(figur 4.6) skjer etter du har trykt på "Add" knappen, dette vil føre deg videre til native funksjoner på din Iphone/Android. Der du vil få muligheten til å legge til det du har valgt. Men det som er mest viktig når du ser på figur 4.5 og 4.6 er at dette ser og føles kjent ut.

Ikoner:

- Valg av ikoner, er gjort ut i fra design systemet til Vizrt.
- Fargen blir valgt ut i frå andre Vizrt plattformer, slik at det skal være en rød tråd gjennom Vizrt sine plattformer.

Plassering ikoner(figur 4.5):

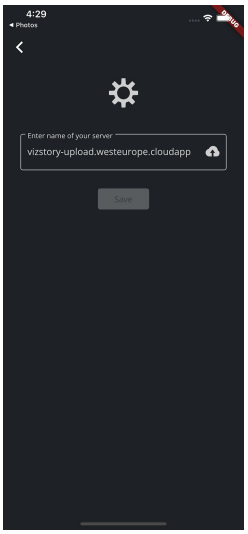
- "Add":
    - Den er større enn de andre for den skal ha hovedfokuset over logen, for det er denne brukerne vil bruke mest.
    - Midt i mellom de 2 andre er den plassert og ikke for lang oppe, for då vil da se ut som det er en varsling.
    - Den er lett og treffe med tommelen, slik som den er plassert no.

  - Når man trykker på denne knappen, så vil du få opp bildet til høyre(figur 4.6). Hvis brukeren har vert bort i ios så kjenner man igjen valg menyen. Som er hele poenget når man får dette valget. Det som er spesielt med fargen er den endrer seg med hvilket farge du har på ditt ios system, om det er light eller dark theme.

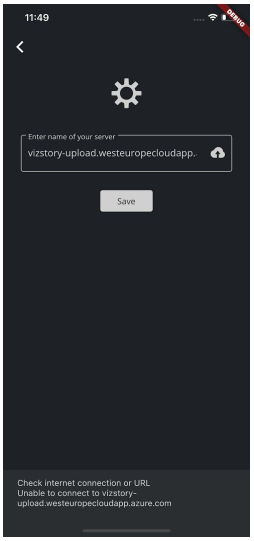
- "Kamera":
  - Den er mer nøytral enn "Add" knappen, for den skal ikkje ta for mye oppmerksomhet, men fortsatt godt synlig.
  - "Kamera" knappen er plassert slik at man kan enkel trykke den med tommelen
  - Når man trykker på knappen så blir overført til native kamera appen

- "Video":
  - Den er også mer nøytral enn "Add" knappen, for den skal ikkje ta for mye oppmerksomhet, men fortsatt godt synlig.
  - "Video" er plassert slik at man kan enkel trykke den med tommelen akkurat slik som "kamera" knappen.
  - Når man trykker på knappen så blir overført til native video appen

- "Settings":
  - Denne knappen er plassert oppe til høyre, slik som de fleste apper har sin settings knapp.
  - Knappen er blitt gjort mindre, men har samme farge. Grunnen for at den er mindre er at den skal ikkje ta fokus i vekk frå knappene nederst.

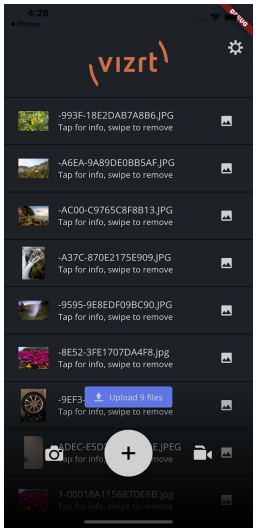


Figur 4.7: Setting screen

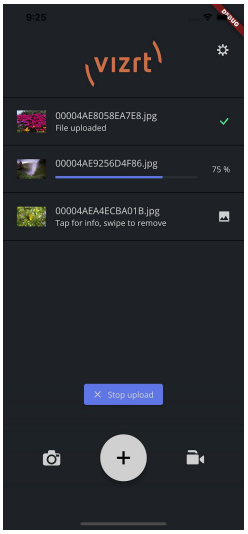


Figur 4.8: Setting screen info

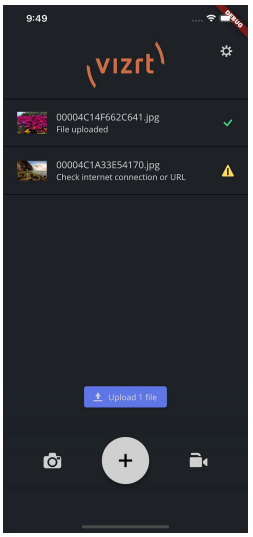
Setting screen(figur 4.7) er veldig enkelt bygd opp, ved at du skriver inn den serveren du vil laste opp til. Hvis man skriver inn en server som ikke fungerer så vil man få en advarsel som man kan se på figur 4.8. Dette designet er hentet i fra den første siden(figur 4.3), slik at du skal kjenne igjen, her skal man skrive inn serveren.



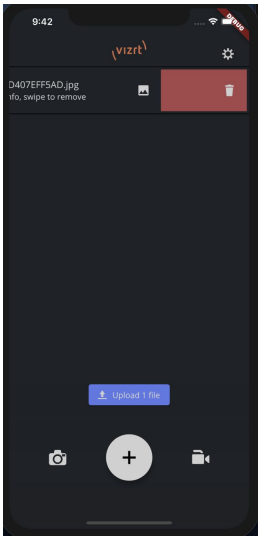
Figur 4.9: Filer



Figur 4.10: Opplasting



Figur 4.11: Advarsel



Figur 4.12: Slett

Filer (Figur 4.9):

- Her ser man hvordan det ser ut når det er blitt lagt til 9 filer, slik at de forsvinner i bakgrunnen.
- Det er blitt lagt til skygge, slik at det ser ut som de forsvinner gradvis ut av skjermen.
- Over "Add" knappen blir det oppretta en ny knapp som har hensikt og vise hvor mange filer du skal laste opp. Det som skjer når trykke på knappen er at den begynner å laste opp til serveren

Opplasting (Figur 4.10):

- Når man trykker på upload knappen, så begynner den og laste opp.
- Når du begynner å laste opp så vil du kunne sjå 2 ting:
- en opplastingsbar som viser fremgangen på opplastingen, på den enkelte fil.
- en % som viser hvor mange prosent framgang du har.
- Du vil få opp grønn "check" som eit bevis på at filen er blitt lastet opp. Som du ser så kan man se at fargen er hentet fra firgur 4.2

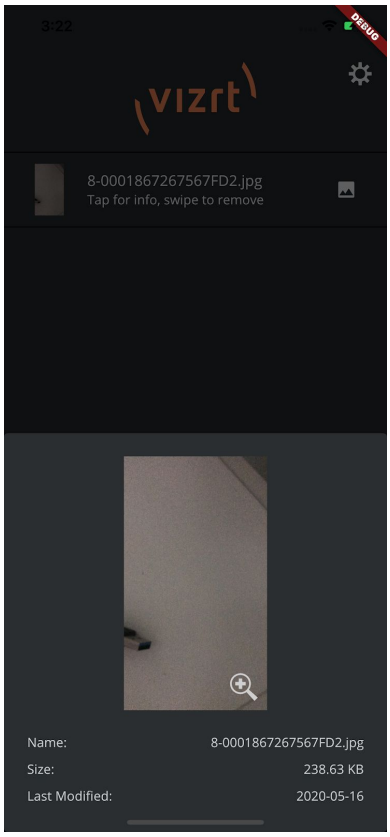
Advarsel(Figur 4.11):

- Det vil vise en varseltrekant om noe er feil
- Mistet internett forbindelsen
- Feil med URL
- Når feilen er rettet opp igjen kan man gjenoppta opplastingen.
- Fargen er hentet fra figur 4.2

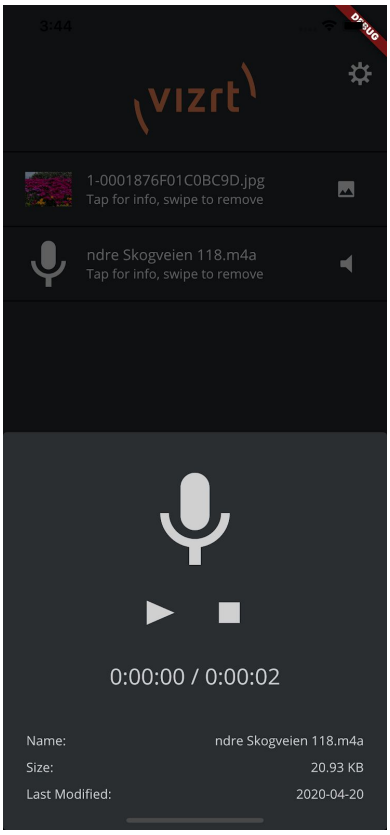
Slett (Figur 4.12):

- Hvis man ønsker og slette/fjerne en fil, så kan man sveipe til en av sidene og filen vil bli slettet.

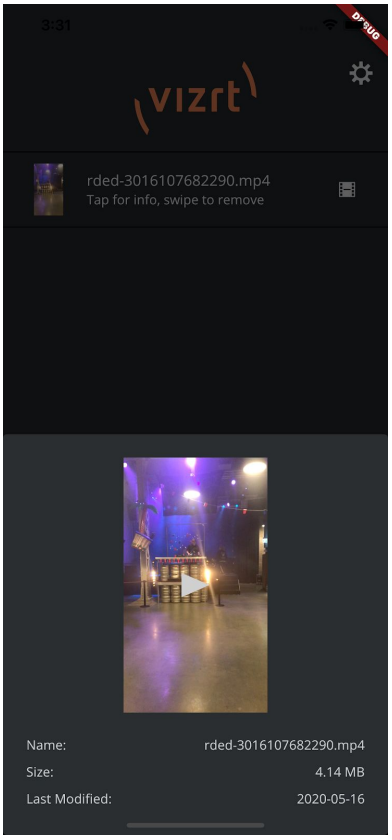
Videre skal vi se preview funksjonen. Denne funksjonen er grei hvis man ønsker å få en oversikt over hva man skal laste opp. Slik at brukeren kan dobbeltsjekke eller ta en ny vurdering på hva som skal laste opp.



Figur 4.14: Bilde



Figur 4.15: Lydfil



Figur 4.16: Video

Her ser du de 3 forskjellige forhåndsvisninger i fra de 3 opplastingsalternativene du har. På alle bildene så får du de samme informasjonen av den filtypen du har tenkt til å laste opp. Men det 3 forskjellige design typer som vi har valgt og bruke.

Bilde(figur 4.14):

- Første tegnet du vil få på at det er et bilde er forstørrelsesglasset nede i høyre hjørnet. Trykket du hvor som helst på bildet så vil det bli større.
- Bilde som blir større kan du sveipe vekk, ved å gjøre det som det føles naturlig.

Lydfil(figur 4.15):

- Her vil man kun se en forstørrelse av ikonet, siden det ikke er et bilde å vise
- Her er det mulighet for og starte og stoppe avspillingen av lydfilen.
- ikoner er valgt ut i fra hvordan det føles naturlig å trykke.

Video(figur 4.16):

- Her vil du se et avspillings icon som betyr at denne filen er av type video og du kan spille denne av ved å trykke på skjermen
- Etter å ha spilt av video, går videoen over til fullskjerm, så kan du sveipe for å gå tilbake til den opprinnelige skjermen

5 EVALUERING

5.1 Evalueringsmetode

Planen var at vi skal få til betatesting av applikasjonen. Vi har fått laget beta på både Android og iOS. Hvor det å få til en beta på på iOS var mye vanskeligere enn på Android. På Android er det bare å lage en apk fil og sende den til de som ønsker å teste applikasjonen. På iOS derimot må man ha en Apple Developer Konto <https://developer.apple.com/programs> , og Vizrt hadde dette, men det at vi kunne få tilgang til denne var ikke bare enkelt. Vi fikk tilslutt tilgang og kunne da lage en betaversjon, men det tok også tid å få denne applikasjonen godkjent av Apple i TestFlight <https://testflight.apple.com> systemet deres i App Store Connect.

Metoden som ble brukt i starten, var at vi laget ukentlige versjoner av applikasjonen slik at de andre på teamet vårt kunne evaluere applikasjonen og se fremgangen. Her fikk vi gode tilbakemeldinger og vi implementerte ønsker og endringer fra Viz Story teamet til neste ukes versjon. Vi har ikke fått tilbakemelding fra noen andre bortsett fra TV2 sin reporter etter Vizrt Story teamet sitt ønske på grunn av en salgsavdeling hos Vizrt som kunne endt opp med å selge et uferdig produkt. Etterhvert som de viktigste funksjonene var på plass, som resulterte i et Minimal Viable Product, kunne eksterne også være med i evalueringen. Jira ble tatt i bruk ca halvveis i prosjektet og vi fikk der en liste med krav av funksjoner som var ønsket implementert fra produkteier. Disse kravene var:

- En skjerm med innstillinger. Denne skjermen skulle ha en komponent til å lagre og hente delte preferanser, som i hovedsak vil si link til service dokumentet til kunden for å koble seg opp til egen instans av Viz Story. Appen skal bruke linken til service dokumentet for å parse og navigere gjennom service dokumentet for å finne URL som appen skal bruke til å laste opp filer til. Selve innstillingsskjermen.
- En funksjon for å bla gjennom galleri og velge medie som skal bli lastet opp, dette fungerte litt forskjellig på Android og iOS så individuelle løsninger krevdes her. Vi skulle lage en oversikt over allerede valgte medier som skal bli lastet opp.
- Dele filer fra andre applikasjoner, si man for eksempel allerede blar gjennom galleriet og ønsker å dele en fil uten å først gå inn på applikasjonen vi utvikler skal man ha mulighet til å gjøre det direkte der man allerede er. Dette innebærer også at man ikke skal kunne dele andre filer med applikasjonen som Viz Story ikke tillater fra før og en støtte for å ta imot mer enn en fil om gangen.

- Lage en brukervennlig varsel om opplastingen feilet. Støtte for å fjerne ferdige opplastninger og legge inn oversikt for å se hvor langt en opplasting er kommet.
- Støtte for bakgrunnsopplasting, altså å lukke ned appen eller låse mobilen uten at appen stopper en påstartet opplasting. Funksjonalitet for å la appen åpne mobilen sitt kamera for å filme eller ta et bilde som så går rett inn på appen.
- Organisere køen for opplastinger som ikke er startet på.

Vi fikk noen fra TV2 som ønsket å teste appen. Vi sendte med en mail om ulike funksjoner i appen som de kan få prøve seg litt med. Vi forventer en enkel tilbakemelding om hva de syntes og deres korte erfaring med mobilapplikasjonen

Vi har ikke hatt mye fokus på testing før slutten ettersom at vi ikke har mye erfaring innenfor testing fra før. Til tross har vi laget 6 enhetstester som leter etter widgeter og ser at disse kjører som de skal. I detalj så pumper vi applikasjonen eller en dummy applikasjon med en spesifikk Widget inn i testen, for så å sette opp en innebygd søker til å finne den aktuelle skjermen. Til slutt setter vi opp hva testen forventer å finne, så for eksempel til HomeScreen forventer testen at den skal finne en “Widget”.

De forskjellige enhetstestene er

- Test app - sjekker Widgeten material app kjører, dette er en applikasjon som bruker material design
- Not arrived at homescreen - en test som leter etter “Homescreen” widget. Denne er her etter et forsøk på å finne ut av hvordan man venter på appen.
- Finds a loading screen - loading screen er den første skjermen som kommer opp når vi kjører mobilappen.
- Test scroll view - Sjekker at NestedScrollView widgeten fungerer, dette gjør at man kan bla i alle filene man har lagt til for opplastning.
- Finds filelist - Dette er listen med filer som man skal laste opp.
- Not arrived at welcomescreen - Denne testen er i teorien den samme som not arrived at homescreen, men er for å være sikker på at lasteskjermen er den første som dukker opp.

Vi har valgt å ikke skrive integrasjonstester på grunn av manglende kunnskap om hvordan man skal skrive en effektiv integrasjonstest. Vi har derimot valgt å heller debugge de forskjellige modulene våre for å forsikre oss at informasjon blir delt på en skikkelig og robust måte mellom modulene våre. Ettersom at applikasjonen må ha en tjener å koble seg opp til, har vi lagt til en test som appen kjører for å forsikre seg om at den er koblet til og får svar fra tjeneren.

5.2 Evalueringsresultat

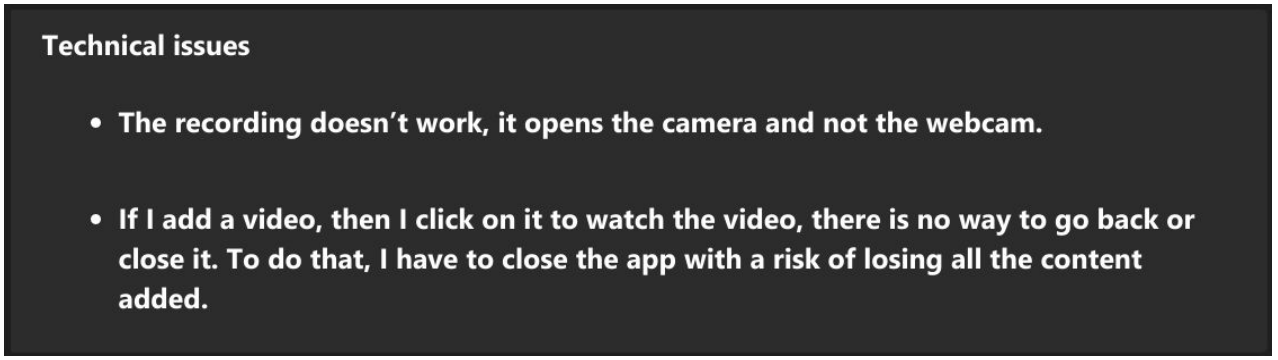
Tilbakemelding fra Vizrt har vært veldig positive. Vi har spurt etter forbedringer helt til det ikke har kommet flere. Vi har også blitt begrenset med brukertesting av Vizrt. De ville ikke dele produktet med for mange innenfor Vizrt eller dele for mye ut av produktet, dette argumenterte de med en veldig pågående salgsavdeling som kunne endt opp med et uferdig produkt. På grunn av dette har vi hovedsakelig hatt en dialog med Vizrt Story team. På det grunnlaget har vi alltid utviklet modulene fram til den muntlige tilbakemeldingen fra Vizrt Story team var at de var fornøyd. Les mer om tilbakemelding fra Vizrt i deres tilbakemelding i appendix 10.5.

Vi fikk gjennom utviklingsprosessen utviklet alle ønskene som Viz Story teamet kom med. Disse ble vist fram og testet kort tid etter de var utviklet. Vi fikk tilbakemelding på hva de syntes om progresjonen og de nye funksjonalitetene til applikasjonen hver uke. Vi følte dette var en veldig bra måte å få tilbakemelding på og fikk raskt gjort endringer om det var ønsket fra tilbakemeldingene vi fikk.

Av integrasjonstestene får bruker opp varsel fra appen når de kobler seg til en VizStory instans eller varsel dersom de ikke får koblet seg til.

Når det kommer til teknisk testing av applikasjonen har vi hatt problemer med å finne ut av Flutter og Dart sitt testebibliotek. Det fungerer fint på det øverste laget av mobilapplikasjonen, men når mobilapplikasjonen begynner å kjøre asynkront begynner problemene å oppstå. Et eksempel på dette er når testen leter etter de forskjellige skjermene finner den bare første skjermen, lasteskjermen. Vi har forsøkt å la testen appen kjøre i 5 minutter for å forsikre oss om at det ikke bare er testen som blir avsluttet for tidlig. Vi finner fortsatt bare lasteskjermen til tross for at vi kan se på Dart Devtool<https://flutter.dev/docs/development/tools/devtools/overview> at enten velkommenskjerm(på første oppstart av app) eller hjemskjermen dukker opp etter kort tid. Vi har hatt problemer med å finne informasjon om hvordan vi burde lage disse testene. Det veldig lite offisiell informasjon om hvordan testene fungerer. Det vi har funnet av informasjon fra upålitelige kilder som Stack Overflow og lignende, har enten ikke passet eller ikke fungert.

Enhetstestene våre er for det meste "Widget"-tester, disse testene ser om appen har en bestemt "Widget" som kjører. En "Widget" er en komponent i Flutter rammeverket, det er byggeklossene i applikasjonen. Her kan vi endre om testen får en suksess eller en feil bare basert på hvor mange plasser testen venter på svar fra appen. Dette er i teorien hva alle enhetstestene våre går ut på. De søker etter forskjellige widgets som vi har i applikasjonen. Eneste enhetstesten vi har som fungerer ser etter en "MaterialApp" widget <https://flutter.dev/docs/development/ui/widgets/material>. Dersom den eksisterer vet vi med sikkerhet at mobilapplikasjonen kjører, men ikke til hvilken grad den kjører. Dette er noe den gjør og har vært veldig grei i bruk med Travis CI for å sjekke om testene er vellykket eller ikke. Et eksempel på hvor det hadde vert greit å ha enhetstester er for å sjekke at appen klarer å åpne kameraet og videokameraet, vi hadde på tidspunktet når vi leverte en demoversjon til en i Vizrt som ikke har vært borti appen en liten bug på grunn av en boolean som var feil verdi på. Dette er noe som en enkel enhetstest kunne plukket opp. Heldigvis var dette som sagt et lett problem å ordne opp ettersom det bare var å endre en boolean verdi.



6 RESULTATER

6.1 Unit testing

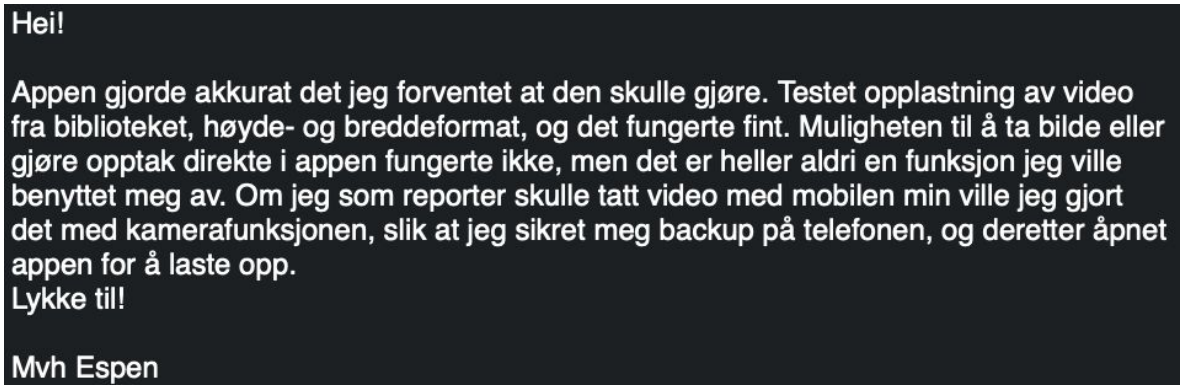
Alle våre unit tester går gjennom som suksess, disse testene er veldig simple og fungerer greit fram til man gjør større endringer på appen. Dersom man skulle ønske å bytte ut noen av widgetene vi tester mot en ny widget som fungerer vil disse testene ikke lenger kjøre som suksess. Disse testene vil forhåpentligvis komme til god nytte når produkteier tar over produkter og skal vedlikeholde og videreutvikle produktet.

6.2 Integrasjonstesting

Vi har gjennom oppbyggingen av prosjektet utført mye testing og debugging på hvordan appen fungerer. Dette for å forsikre oss om at all informasjon som trenger å bli sendt mellom moduler blir sendt og kommer dit den skal. Med andre ord at appen fungerer som den skal. Hver modul har derfor blitt debugget til stor grad ettersom de på et tidspunkt ikke har fungert. Dette for å forsikre at all informasjon kommer fram og at det ikke skal dukke opp noen uforutsette problemer. Vi har for sikkerhets skyld lagt inn en test som appen kjører når man linker appen opp mot et Viz Story instans, så dersom bruker skriver inn feil link eller Viz Story instanset ikke kjører eller eksisterer vil de få opp varsel om at de ikke er koblet til. Likt om de kobler seg til en Viz Story instans vil de få opp en varsel om at de er koblet opp til en instans.

6.3 Brukertesting

Vizrt har fått en Journalist fra TV2 til å teste mobilapplikasjonen. Det å få en i målgruppen til å teste mobilapplikasjonen er uvurderlig og passer oss helt perfekt. Tilbakemeldingen er som følger:


Hei!

Appen gjorde akkurat det jeg forventet at den skulle gjøre. Testet opplastning av video fra biblioteket, høyde- og breddeformat, og det fungerte fint. Muligheten til å ta bilde eller gjøre opptak direkte i appen fungerte ikke, men det er heller aldri en funksjon jeg ville benyttet meg av. Om jeg som reporter skulle tatt video med mobilen min ville jeg gjort det med kamerafunksjonen, slik at jeg sikret meg backup på telefonen, og deretter åpnet appen for å laste opp.
Lykke til!

Mvh Espen

Det tilbakemeldingen sier her er at vi har noe problemer i koden vår, i tillegg har vi implementert en funksjon i applikasjonen som ikke nødvendigvis kommer til å bli brukt.

Vi har også fått tilbakemelding fra Viz Story teamet som har mye erfaring allerede, men ikke innenfor mobilapplikasjonutvikling. Deres tilbakemelding er fortsatt veldig viktig for oss. De har gitt oss mye viktig tilbakemelding så vi kunne rette på dette. De siste demoversjonen vi har gitt teamet har de vært veldig fornøyd med.

I 10.6 kan du lese tilbakemelding fra brukertesting. Her har de kommet med tilbakemelding om noen funksjoner som ikke har fungert skikkelig, men som har blitt rettet opp i på kort tid. De har også ønsket en måte å lagre bilde og video på mobilen, som er blitt tatt via appen, som en måte å backe opp filen. De har påpekt begrensninger vi har støtt på i flutter sin støtte mot iOS. Punkt 6 og 7 i tilbakemeldingen gir de forslag på enklere måter som fillisten kan oppføre seg på som er veldig nyttige for videre utvikling av appen. I tillegg til et ikon som egentlig bare er til design som har vært forvirrende.

6.4 Akseptansetesting

Oppgaven vi fikk fra Vizrt var å lage en mobilapplikasjon for å laste opp bilde, film og lydfiler til en Viz Story instance kjørende på en server. Løsningen vår gir brukeren mulighet for at appen åpner kamera, videokamera og galleri for enten ta bilde, filme, eller velge fil som eksisterer fra før for å laste opp til serveren. Brukeren kan bytte rekkefølge på filene som skal bli laste opp, avbryte pågående opplastning, få en preview av bilde, film og lydfil i mobilappen og fjerne fil som er gjort klar for opplastning. Brukeren kan også koble seg opp til sin server ved å skrive inn en enkel link. Deretter finner mobilappen service dokumentet og parser den informasjonen for å finne den rette URL'en å koble til.

7 DISKUSJON

I denne oppgaven valgte vi å lage en kryssplattform applikasjon med rammeverket Flutter med Dart, vi valgte å bruke SCRUM sammen med Kanban i utviklingsprosessen og vi valgte å prøve å lage en prototype raskest mulig slik at vi kunne raskt få tilbakemeldinger fra Vizrt. Vi hadde sprinter på 1 uke hvor vi hadde demo av hvor langt applikasjonen var kommet hver mandag sammen med teamet vårt fra Vizrt.

Vi valgte å ikke gå for en TDD, "Test driven development" (Tester først utvikling) , strategi i dette prosjektet. En av grunnene til det er at ingen av oss er særlig drevne i slik utvikling. Vi ønsket også å raskt få til en prototype slik at vi kunne få tilbakemeldinger fra Vizrt. Det var til stor hjelp for oss, da både vi og Vizrt var en del usikre på designet av applikasjonen, og vi kunne da tidlig begynne å bevege oss i en design retning og finjustere den. En annen grunn var at vi hadde tidsbegrensning på prosjektet og Vizrt var heller ikke helt sikre på all funksjonalitet som skulle i applikasjonen fra start. En negativ side med at vi ikke gikk for en TDD var at flere bugs mulig har blitt og ble introdusert mens vi utviklet applikasjonen. Et eksempel var da vi sendte ut en beta til noen testere, og de rapporterte tilbake at de ikke hadde mulighet til å filme i applikasjonen. Løsningen var å endre en enkel variabel verdi i koden. Det er mulig at slike problemer kunne vi unngått med TDD, noe som hadde mulig resultert i en mer robust kode. Men fordelen med at vi ikke gikk for TDD er at Vizrt og vi kunne mye enklere redefinere kravspesifikasjonene til applikasjonen uten at vi hadde gjort mye arbeid med å skrive tester til funksjonalitet det senere ble bestemt at vi ikke hadde tid til å implementere.

I de siste par ukene av utviklingen har vi skrevet noen tester for mobilapplikasjonen. Vi har siden begynnelsen hatt noen enkle tester bare for å sjekke at appen kjører. Vi har så bestemt oss for å lage flere tester for å sjekke at de forskjellige enhetene og widgetsene fungerer som de skal. Når vi skulle først utvide klassen med tester dukket det opp problemer med å finne widgets som vi visste eksisterte. I begynnelsen funderte vi på om dette hadde noe å gjøre med den asynkrone oppførselen til applikasjonen. Vi har testet masse forskjellig og prøvd å vente på svar fra applikasjonen på flere forskjellige steder. Blant funnene har vi oppdaget at den testen går gjennom til tross for at den både finner og ikke finner en bestemt widget, noe som skal være umulig. Etter mer testing har vi funnet ut at resultatet fra testene varierer etter hvor vi venter på svar fra mobilapplikasjonen. Svarene fra flere av testene er ulogisk og ikke sammenhengende. Dette gjorde prosessen med å finne ut hva som er riktig gjennom logisk prøving og feiling på testene veldig

vanskelig. På grunn av dette har vi lagt testingen litt på hyllen. Her har altså Flutter noe å jobbe med å levere bedre informasjon på hvordan man skal teste asynkrone applikasjoner. Et eksempel på noe som ville vært unngått om vi hadde hatt mer fokus på tester er en fra Vizrt som har testet appen som fikk problemer med noe simpelt som vi kunne fått en test til å sjekke på. Dette var heldigvis en veldig liten feil å ordne opp i, men som kunne vært unngått.

Når det kommer til faktisk brukertesting har det egentlig bare vært i den siste uken at vi fikk noen reportere til å teste applikasjonen. Det som har tatt en del tid er å få applikasjonen lastet opp til Apple sitt testprogram "Test Flight", da Apple må godkjenne applikasjonen før den kan sendes ut til eksterne testere. Vi visste ikke at det kom til å ta så lang tid som det tok, da Apple visstnok også måtte få en url til en server slik at de kunne teste alle funksjoner av applikasjonen før de lot oss sende den ut. Det viste seg å ta opp til en uke å få godkjent, så det var noe vi burde ha begynt med litt tidligere i løpet slik at vi kunne hatt flere uker å teste applikasjonen på iOS.

En tilbakemelding i fikk fra en reporter gikk på muligheten til å ta bilde og video direkte i applikasjonen. Han mente selv at dette var en funksjon han aldri ville benyttet seg av. Han forklarte dette med at om han skulle ta et videoopptak med mobilen så ville han heller ha brukt den innebygde kamera-applikasjonen på telefonen, slik at han kunne ha sikret seg en backup, og deretter åpnet applikasjonen vår for å laste videoen opp. Dette er noe vi kanskje burde ha funnet ut på et mye tidligere tidspunkt. Når vi startet å utvikle applikasjonen gikk vi ut fra Vizrt sine kravspesifikasjoner uten å utføre noen som helst markedsanalyse eller høre direkte med noen reportere som faktisk skulle bruke applikasjonen om hvilke funksjoner de ønsket eller kunne tenkt seg i applikasjonen. Hadde vi gjort det kunne vi på et mye tidligere stadie finne ut hvilke funksjoner i applikasjonen som faktisk var nødvendige å implementere, og kanskje også ha tid til å legge til noen andre funksjoner. Alternativt kunne vi ha mye tidligere ha gitt ut en beta til denne reporteren slik vi var inne på i forrige avsnitt.

Noe som også kan diskuteres er hvordan resultatet på applikasjonen hadde blitt om det ikke hadde vært en pågående pandemi i verden. Da hadde vi kunnet jobbet sammen på kontoret til Vizrt, som originalt var planen. Etter regjeringen stengte ned landet, ble all kommunikasjon digitalisert. Dette kan ha ført til flere ting, og kan vært vanskelig å måle nøyaktig i hvor stor grad situasjonen har påvirket prosjektet vårt. Noe som naturligvis ble vanskeligere var å vite hva som ble gjort på applikasjonen til en hver tid, når vi ikke kan ha

møte eller snakke sammen hele dagen. Løsningen vi hadde på kommunikasjonsproblemet så oppsto ved at vi ikke kunne jobbe i samme rom, var å hyppige videomøter. I forhold til Vizrt hadde vi statusmøte hver mandag, hvor de kunne få innblikk i progresjonen og hvor vi kunne stille spørsmål om det var noe vi lurte på. Vi hadde noe hyppigere møter oss i mellom, omtrent 2 ganger i uken, hvor vi fordelte arbeidsoppgaver og delte problemer vi støtet borti. Strategien vi valgte, SCRUM sammen med Kanban, gjorde at vi kunne holde til en viss grad en oversikt over hvem som gjorde hvem og fremdriften i prosjektet. Det er mulig at denne strategien har vært den beste i denne situasjonen for den har ført til at vi og Vizrt hele tiden har kunnet holde en kontroll på hva som må gjenstår å gjøres fra initielle krav, hva som var underveis og hva som var blitt gjort.

Det viste seg at Flutter på mange måter er et brilliant rammeverk for kryssplattform applikasjoner, men det fantes selvsagt ulemper. Noe av det som var vanskeligst i vårt tilfelle var hvordan vi kunne legge til filer i applikasjonen. Måten du kommuniserer med filsystemer er forskjellig på iOS og Android, og det resulterte i at vi også måtte ha to forskjellige måter å hente inn filer på, Her brukte vi også en plugin som var utviklet til dette formålet, da Flutter ikke hadde noen innebygd måte å vise en filutforsker på, siden dette er noe som sagt er forskjellig på ulike plattformer. Dette førte til et par begrensninger, som for eksempel antall bilder det var mulig å velge samtidig på iOS, grunnen til dette var at filutforskeren bare gikk til Filer-mappen på iOS og ikke bilde og video galleriet. Måten vi løste dette på var å ha en annen plugin i tillegg til å vise galleriet og gi brukeren et valg i applikasjonen om hvor den ønsket å hente medieinnhold fra. Men det var her at pakken som ga brukeren tilgang til galleriet hadde en begrensning på at man bare kunne velge et bilde av gangen. For å kunne legge til lydfiler til applikasjonen måtte vi åpne opp for at alle type filer kunne bli valgt, selv filer som ikke var støttet å bli lastet opp til Viz Story, vi løste dette med at om en ustøttet fil ble lagt til i applikasjonen vises en melding i applikasjonen om at kun bilder, video og lydfiler støttes og at vi ikke legger til filen i det hele tatt. Filutforskeren var derimot veldig fungerende på Android, brukeren kunne velge filer både lokalt på telefonen og fra kontoer i skyen som for eksempel Google Foto. Det eneste problemet her var at bilder som ikke lå fysisk på telefonen hadde jo selvsagt ingen filplassering og filnavn på telefonen, så filen ble mellomlagret og kunne bli vist i applikasjonen, men det var ingen måte å vite hva filtypen var, da denne mellomlagringen bare hadde en id som filnavn uten noe filetternavn. Derfor måtte vi her også på Android gi brukeren valget om hvilken filtype de ønsket for så å bare vise denne filtypen slik at applikasjonen kunne være sikker på hva type filen de fikk var slik at den kunne behandle den riktig.

Flutter er enda i et veldig tidlig stadie, noe vår erfaring med rammeverket viser. Det kunne ført til noen begrensninger i hva applikasjonen var i stand til å gjøre. Vi støttet ikke på noen særlig begrensninger på hva som var mulig å gjøre i selve applikasjonen, men det var med en gang vi begynte å bevege oss utenfor applikasjonen og begynte å snakke med plattformene, for eksempel med filsystemet, at vi støttet på noen problemer.

På den positive siden har Flutter hatt vidunderlige verktøy til å hjelpe med utviklingen. Disse kommer i form av feilsøkingsverktøy og designhjelpere.Videre er vi veldig fornøyde med Flutter sitt enorme bibliotek av ferdige komponenter som kan brukes når man utvikler applikasjoner. Man kan i tillegg gjøre sine egne endringer på disse om det er ønskelig, men veldig mye bare fungerer til å begynne med. Det har ført til at vi har kunnet lage et ganske fint design på appen helt fra start, og vi har kunnet fokusert mer på funksjonalitet enn på hvordan applikasjonen ser ut. Dette er et området hvor Flutter virkelig skinner. For hvor andre kryssplattform rammeverk sliter med store forskjeller på hvordan appen er ut og oppfører seg på ulike plattformer, vil Flutter som standard være “pixel perfect” som de kaller det siden det i bunn og grunn er maskinkode som kjøres direkte på plattformen. I tillegg har Flutter et veldig bra bibliotek for animasjoner, som gjør det veldig enkelt å legge til implisitte animasjoner rundt om i applikasjonen.

Det Flutter i hovedsak har hjulpet oss mest med er å kunne ha en samlet kodebase for hele applikasjonslogikken. Det er slik at man måtte inn i det spesifikke prosjektet for hver plattform for å ordne ting som navnet, id og applikasjonsikon individuelt, men det måtte vi uansett ha gjort om vi hadde gått for en ikke kryssplattform løsning. Det verktøyet vi har likt best med å bruke Flutter er “Hot Reload” funksjonaliteten. Det har gjort det å endre på små detaljer i koden enormt mye enklere enn om vi måtte ha ventet et minutt mellom hver minste endring for å se hvordan det ser ut. Det har ført til at vi har kunnet gjøre mange små finjusteringer i designer som å flytte et ikon 5 piksler mer til venstre, noe vi eller kanskje ikke hadde hatt tid til. Vi mener at Flutter har bidratt stort til hvor bra designet har blitt. Det samme gjelder hvor likt det det ser ut på tvers av de ulike plattformene. Konsekvensene av at vi valgte Flutter og Dart til å lage applikasjonen har med andre ord ikke vært så store.

8 KONKLUSJON OG VIDERE ARBEID

Prosjektets mål var å lage en applikasjonen som kunne forbedre arbeidsflyten til reportere. De ønsket en enklere og mer effektiv måte å få medieinnhold de hadde på mobiltelefonen sin til webapplikasjonen Viz Story, enn den løsningen som allerede fantes. Vi har iløpet av dette prosjektet klart å lage en mobilapplikasjon som gjør denne jobben. En reporter kan enten velge å ta bilde eller filme direkte i appen, eller legge til filer som allerede finnes på telefonen og laste de direkte opp til Viz Story.

Om vi ser på evalueringen og resultatene fra kapittel 5 og kapittel 6 så har vi svart på oppgaven og utført de kravene som var ønsket av Vizrt. Blant annet har Mikal H Henriksen som prosjekteier sagt at de er veldig imponert og sluttproduktet er over forventet. Vi har fullført alle krav Vizrt la fram til oss på Jira og blitt godkjent av Vizrt Story team. Hovedoppgaven var å få laget en app som kan laste medier til en egenvalgt instans av Viz Story. Vi har fikk tildelt en testinstans som vi har testet veldig mye på gjennom prosjektet, både forskjellige filtyper, mange filer og bare enkeltfiler. I tillegg har vi gått gjennom alle delene av appen for å forsikre oss at informasjonsflyten fungerer som den skal. Vi har ikke fått noen tilbakemelding på noen problemer som ikke er blitt ordnet.

I brukertestingen fikk vi litt tilbakemelding, der to av disse var viktige problemer som ble ordnet opp i med en gang og ett problem som vil være en ny funksjon som vi ikke har tenkt på i det hele tatt. Dette kommer godt med videre utvikling av applikasjonen. De siste tilbakemeldingene var for det meste “quality of life” funksjoner som også kommer veldig godt med. Det var også et par designendringer vi kunne endret på. Vi tar godt imot disse tilbakemeldingene og bemerker oss at disse er ikke av noen veldig stor grad, og at disse kunne vært tilbakemeldinger på en allerede utgitt applikasjon.

Videre utvikling av applikasjonen kan inkludere flere funksjoner i forhold til brukeropplevelse. Kanskje applikasjonen kan lagre hva den enkelte personen har lastet opp, eller kunne endre på filnavnet til filen som blir lastet opp. Det kunne også bli lagt til en slags autentisering i applikasjonen slik at man må ha en bruker for å kunne bruke applikasjonen. Dette vil kunne gi det enkelte selskapet som bruker applikasjonen muligheten til adgangskontroll og vite hvem som laster opp hva og muliggjøre et slags eierskap til de opplastede filene. Det hadde da vært en forutsetning av Viz Story webapplikasjonen kunne støttet disse funksjonene.

9 REFERANSER

Google, Dart dokumentasjon, 2020. Hentet fra: https://dart.dev

Lastet ned: 15.05.2020.

Google, Flutter, 2020. Hentet fra: https://flutter.dev
Lastet ned: 29.05.2020.
Google, Flutter dokumentasjon, 2020. Hentet fra: https://flutter.dev/docs
Lastet ned: 02.04.2020.
Google, Flutter dokumentasjon, Platform Channels, 2020. Hentet fra:
https://flutter.dev/docs/development/platform-integration/platform-channels
Google, Flutter dokumentasjon, "Hot Reload", 2020. Hentet fra:
https://flutter.dev/docs/development/tools/hot-reload
Lastet ned: 29.05.2020.
Google, Flutter dokumentasjon, Kompilering, 2020. Hentet fra:
https://flutter.dev/docs/testing/build-modes
Lastet ned: 29.05.2020.
Google, Flutter dokumentasjon, "MaterialApp", 2020. Hentet fra:
https://flutter.dev/docs/development/ui/widgets/material
Lastet ned: 29.05.2020.
Google, Plugins til Flutter og Dart - Pub, 2020. Hentet fra: https://pub.dev
Lastet ned: 27.05.2020.

Google, Dart DevTools, 2020. Hentet fra:
https://flutter.dev/docs/development/tools/devtools/overview

Lastet ned: 01.06.2020.

Microsoft, Visual Studio Code, 2020. Hentet fra: https://code.visualstudio.com
Lastet ned: 27.05.2020.

Adobe, Adobe XD, 2020. Hentet fra: https://www.adobe.com/no/products/xd.html
Lastet ned: 27.05.2020.
Vizrt, Viz Story, 2020. Hentet fra https://www.vizrt.com/products/viz-story
Lastet ned: 02.04.2020.
Vizrt, About Vizrt, 2020. Hentet fra https://www.vizrt.com/vizrt
Lastet ned: 03.05.2020.
Vizrt, MAM, 2020. Hentet fra https://www.vizrt.com/digital/asset-management
Lastet ned: 27.05.2020.

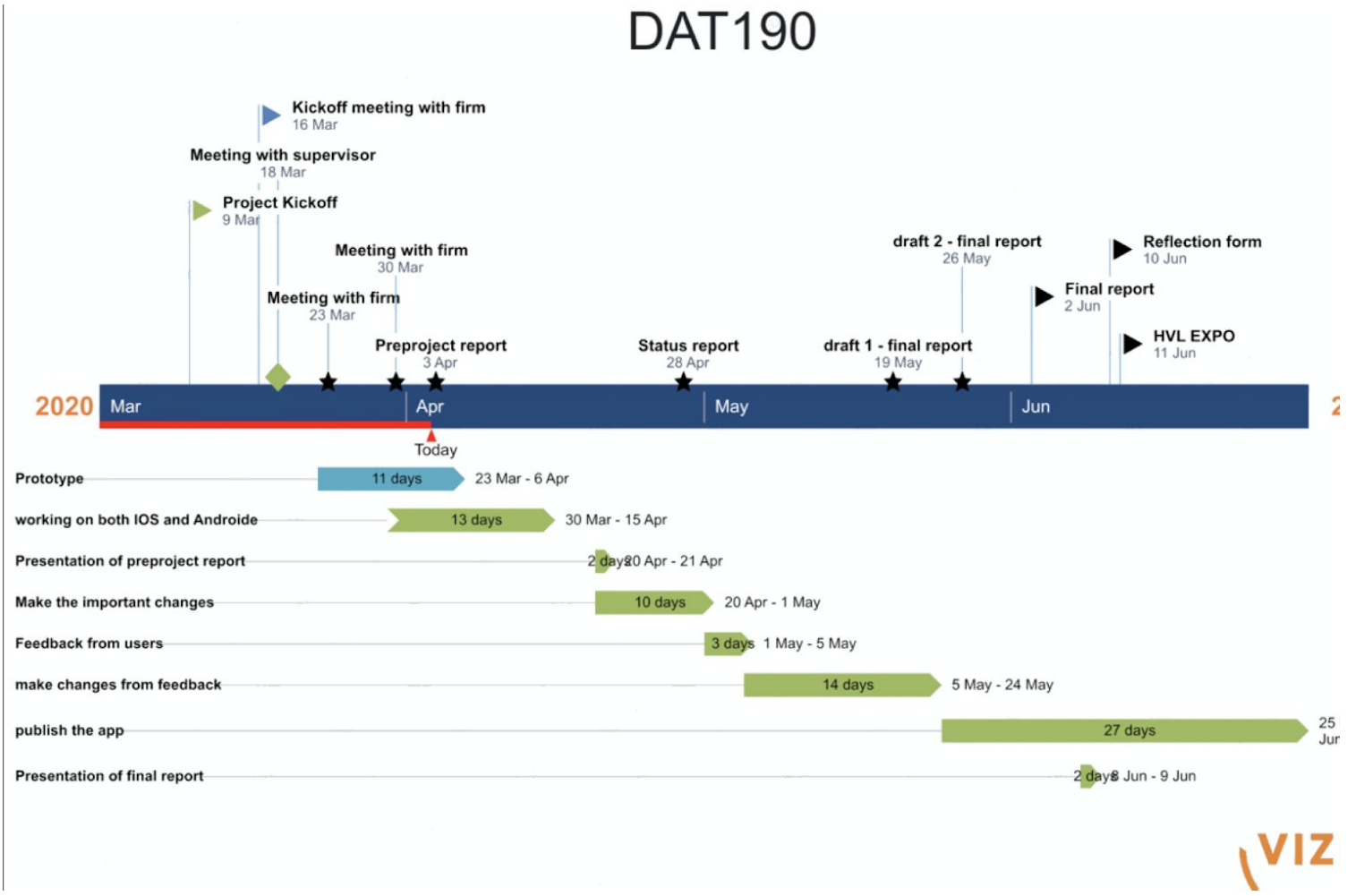
Atom, Protocol, 2020. Hentet fra http://www.atomenabled.org/developers/protocol/#whatIsAtom
Lastet ned: 03.05.2020.
Atom, IETF, 2020. Hentet fra https://datatracker.ietf.org/wg/atompub/about
Lastet ned: 03.05.2020.
Google, Material Design, 2020. Hentet fra https://material.io/design
Lastet ned: 03.05.2020.
Facebook, React Native, 2020. Hentet fra https://reactnative.dev
Lastet ned: 15.05.2020.
Facebook, React Native, "Bridge", 2020. Hentet fra https://reactnative.dev/docs/native-modules-ios
Lastet ned: 29.05.2020.
Microsoft, Xamarin, 2020. Hentet fra https://dotnet.microsoft.com/apps/xamarin
Lastet ned: 15.05.2020.
Ionic, Ionic Framework, 2020. Hentet fra https://ionicframework.com
Lastet ned: 15.05.2020
Apple, Swift, 2020. Hentet fra https://developer.apple.com/swift
Lastet ned: 27.05.2020.
Google, Kotlin, 2020. Hentet fra https://developer.android.com/kotlin
Lastet ned: 27.05.2020.
Nordic Capital, Nordic Capital Fund VIII, 2020. Hentet fra https://www.nordiccapital.com/about/funds-vehicles/fund-vii
Lastet ned: 29.05.2020.
Travis CI, Travis CI, 2020. Hentet fra https://travis-ci.org
Lastet ned: 29.05.2020.
Apple, Apple Developer Program. Hentet fra https://developer.apple.com/programs
Lastet ned: 29.05.2020.
Apple, TestFlight. Hentet fra https://testflight.apple.com
Lastet ned: 29.05.2020.

10 APPENDIX

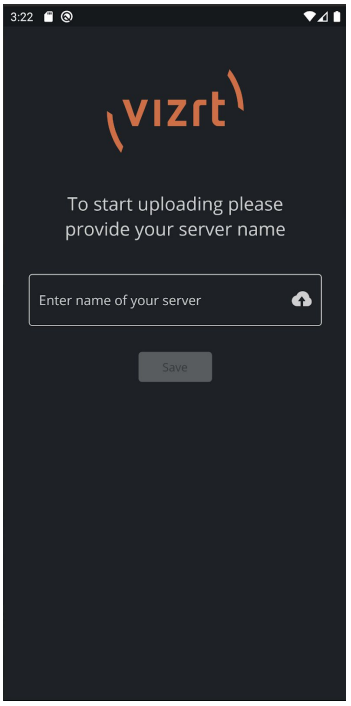
10.1 Risikoliste

Risiko	Alvorlighetsgrad	Sannsynlighet	Mulig tiltak.
Covid-19	Høy	Høy	Jobber hjemmefra og får ikke lov til å dra på kontoret.
Dårlig kommunikasjon	Middels	Liten	Ukentlig møte med bedriften og daglig møte med gruppa.
Koden blir sletta	Høy	Liten	Har flere backuper.
Endring av kravspesifikasjon	Middels	Høy	God dialog, og kunne si nei til endringer som vil ta for lang tid.
Misforståelse i forhold til kravspesifikasjon	Høy	Middels	God kommunikasjon.
Ikke oppnå ønsket produkt på grunn av dårlig tid	Middels	Middels	Være bevisst på hva sluttmålet er og prioritere oppgaver riktig.
Mangel på kompetanse	Middels	Middels	Holde oss oppdatert og si nei til for høye krav. Lære oss det som kreves
Problem i gruppa	Høg	Lav	Flinke til og gi tilbakemeldinger og gi beskjed hvis det er noe.

10.2 GANTT diagram



10.3 Brukermanual



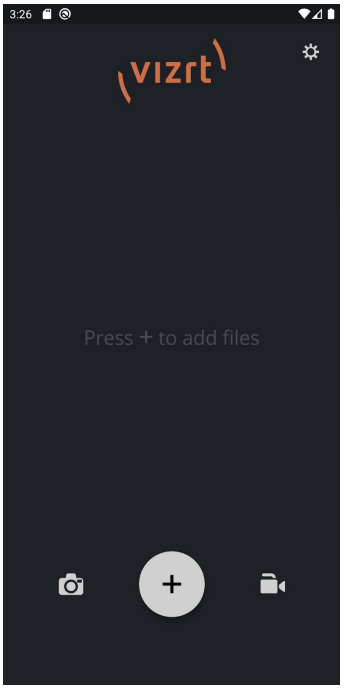
Welcome screen:

- Will only show the first time you try the app
- Enter the url you want to upload to
- A test url is:

https://vizstory-upload.westeurope.cloudapp.azure.com/

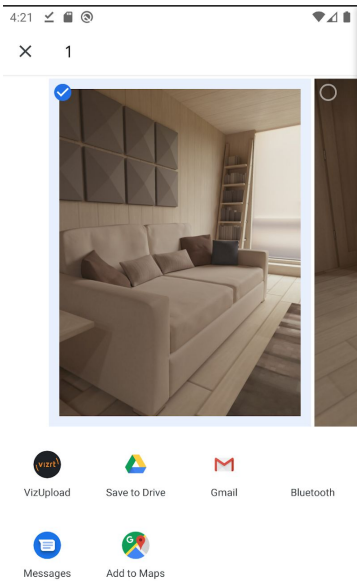
Start screen:

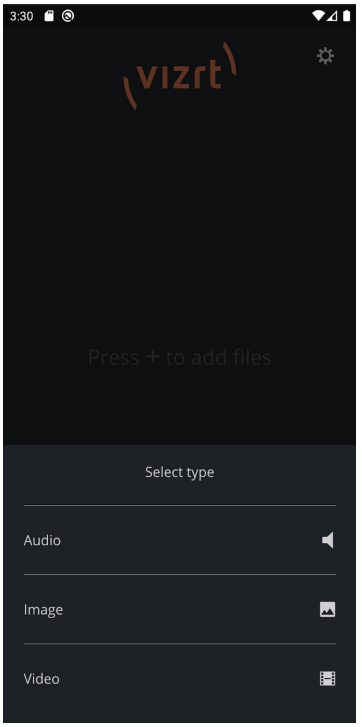
- Press the camera icon to enter your phone's camera viewfinder to take a picture that will automatically show up in the app.
- Press the video recorder icon to enter your phone's camera viewfinder to take a video that will automatically show up in the app.
- Press + to select some media already on your phone.



Choose media type:
(Will look slightly different on Android and iOS)

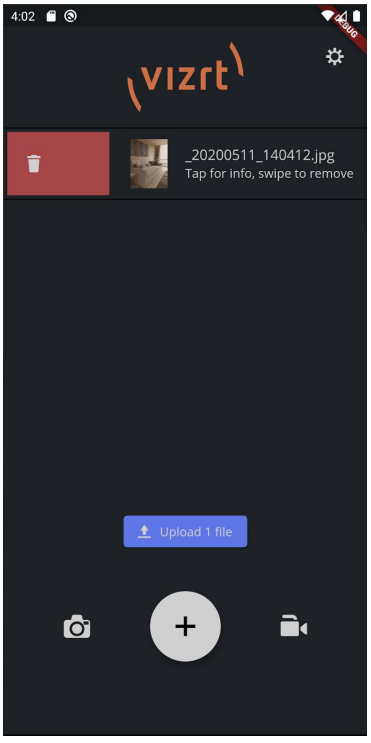
- Select what media you want to add to the app, this will take you to your phone's storage, where you can select files with the specified media type.
- On iOS the choice is between selecting photos or video from the gallery or to choose any files from the Files app.
- You can also share media with the app using the specific share interface on either Android or iOS (here showing on Android)

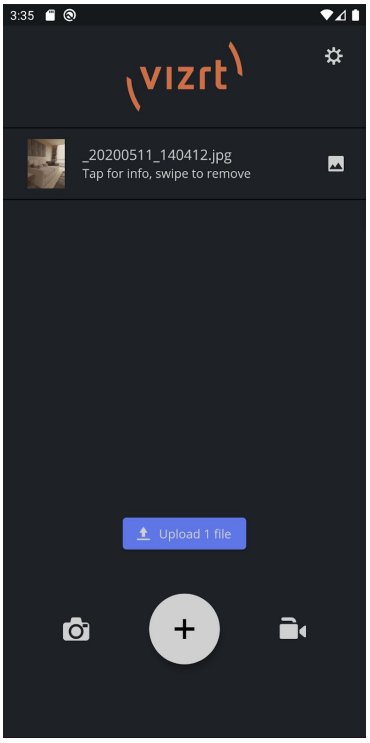




View select files:

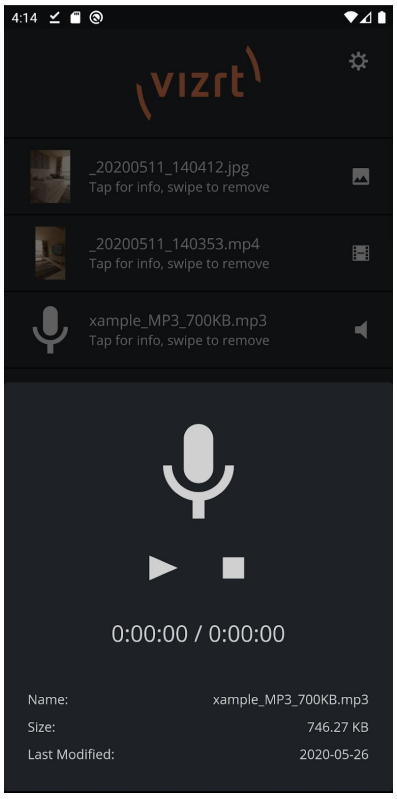
- After selecting desired files to upload, the selected media is presented in a list.
- A blue button will appear showing the number of files to be uploaded, click to upload the files in the list.
- You may add more files to the list at any time.
- The files will be uploaded in sequential order starting from the top of the list. If you want to change the order, you can tap and hold the file you want to move and drag it downwards og upwards.
- If you don't want to upload a file in the list, you can swipe it either to the right or left to dismiss it, you may do this at any time, even after you have started uploading.

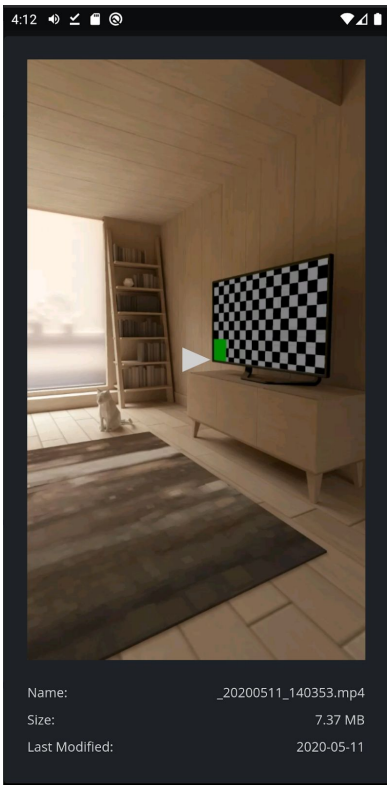


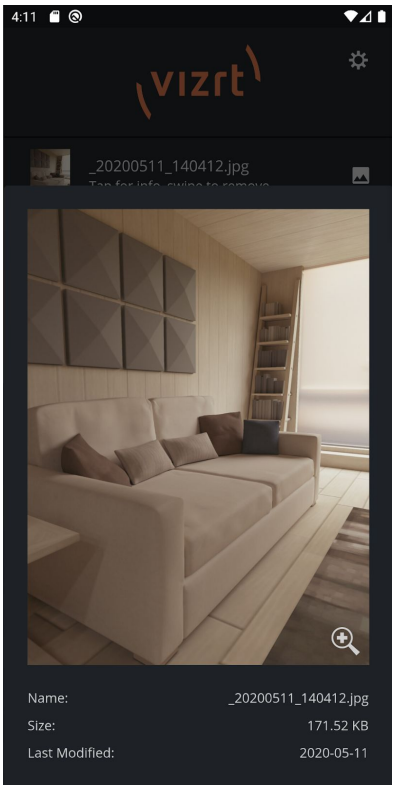


View file:

- You can view any file and information about the file in the list by tapping on the file.
- You can view the image larger with zooming capabilities, watch the video or listen to the audiotrack by tapping further on the icons.
- If you want to go back, when you are viewing a video or picture, you can swipe down.

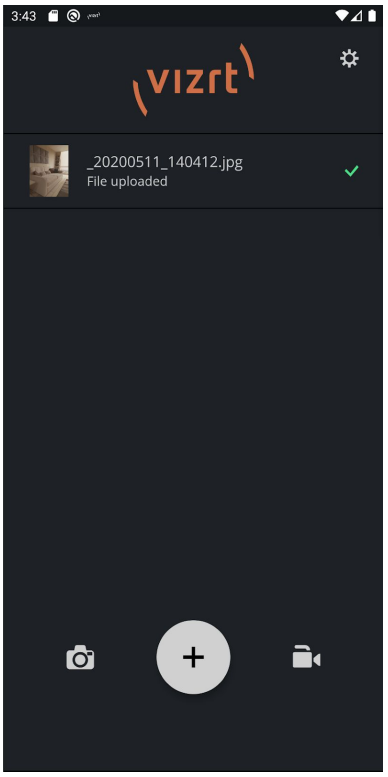


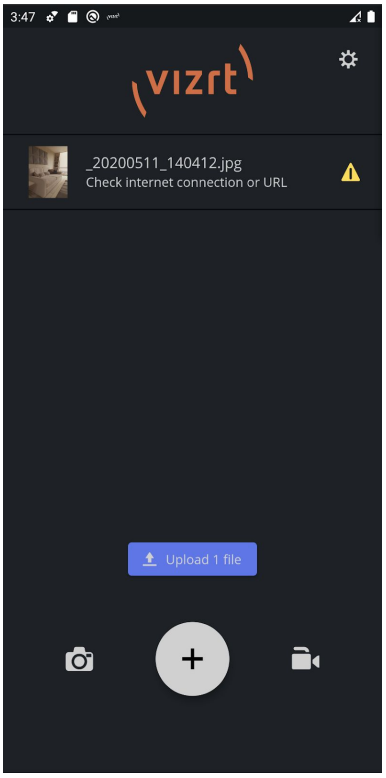




Upload status:

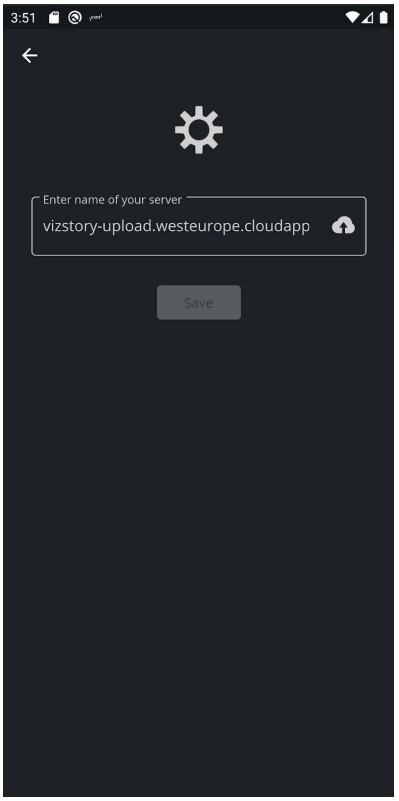
- After pressing the upload button, a progress bar will appear indicating the upload progress of the file.
- If the upload is successful a green icon and a describing text will show to tell you that the upload is completed.
- If any error occurs during the upload, an error message will appear with an appropriate icon, describing the error, you may at any time retry the upload.





Settings screen:

- You may at any time change the upload url to upload to a different server



10.4 Atom Feed

I følge nettsiden atomenabled.org er Atom:

"Atom is the name of an XML-based Web content and metadata syndication format, and an application-level protocol for publishing and editing Web resources."

Et eksempel med Atom feed:

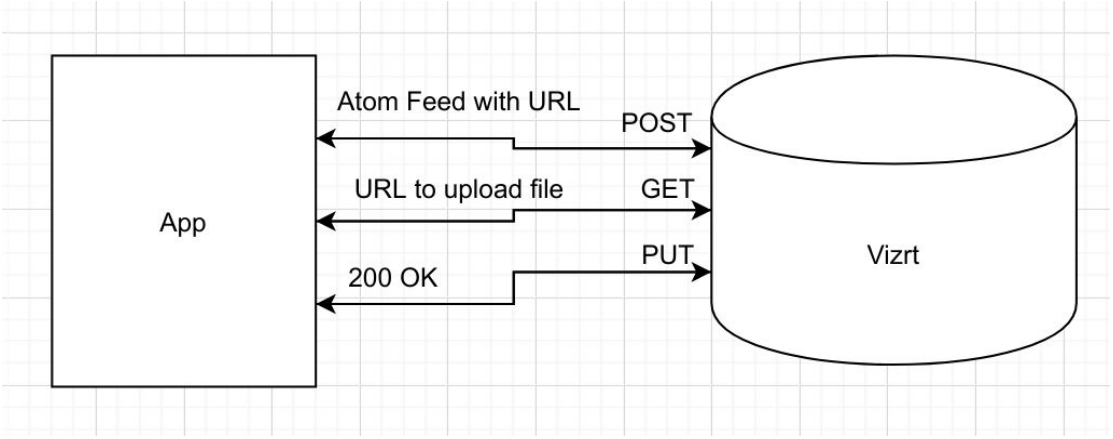
```
<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom">

  <title>Example Feed</title>
  <link href="http://example.org/"/>
  <updated>2003-12-13T18:30:02Z</updated>
  <author>
    <name>John Doe</name>
  </author>
  <id>urn:uuid:60a76c80-d399-11d9-b93C-0003939e0af6</id>

  <entry>
    <title>Atom-Powered Robots Run Amok</title>
    <link href="http://example.org/2003/12/13/atom03"/>
    <id>urn:uuid:1225c695-cfb8-4ebb-aaaa-80da344efa6a</id>
    <updated>2003-12-13T18:30:02Z</updated>
    <summary>Some text.</summary>
  </entry>

</feed>
```

Forklaring på taggene:

- Første linje er standard XML syntax
- <feed> er selve Atom Feeden som henter navnerommet Atom
- <title>, <link>, <updated>, <author> og <id> er metadata om feeden
- <entry> er filen og metadata om den, <link> er linken til selve filen.

The Atom Publishing Protocol som det formelt heter er laget av IETF AtomPub Working Group og det er med denne protokollen mobilapplikasjonen likt som webapplikasjonen skal kommuniserer med serveren. Protokollen går ut på at man først sender en initiell "entry" post forespørsel til serveren. Serveren responderer med en annen url som svar på mottatt initiell forespørsel. Serveren har da altså "gjort plass" til at en ny mediefil kan bli lastet opp. Applikasjonen må så gjøre en get forespørsel på denne mottatte urlen hvor da serveren svarer med å gi en url applikasjonen kan til slutt laste opp mediefilen ved hjelp av en put forespørsel.

Diagram som viser Atom Publishing Protocol:



10.5 Tilbakemelding fra Vizrt

vizrt

Oppsummering av studentprosjekt 'Mobile Upload'

Andreas Garvik, Morten Helland og Sondre Gjesdal valgte å jobbe med prosjektet 'Mobile Upload' sammen med Story teamet i Vizrt. Dette er oppgaven slik den ble presentert i forkant av pitch ved høyskolen i januar:

> *Viz Story en en videoredigeringsløsning som kjører i nettleseren din, laget for å kunne hente inn media, redigere, legge på grafikk og publisere video så raskt og effektivt som mulig på flest mulig kanaler. Viz Story brukes av journalister fra flere store mediehus rundt om i verden, og video publisert fra Viz Story blir vist i alt fra nyhetssendinger i beste sendetid til nettaviser og Twitter og Facebook-poster.*
>
> *Vi mangler i dag en løsning for å kunne enkelt og raskt kunne laste opp video og bilder direkte fra mobil til Viz Story uten å gå veien om å først lagre det på en PC. På samme måte som Viz Story er uavhengig av operativsystem eller nettleser må en mobilapplikasjon være tilgjengelig for både Android og iOS. Det er mye funksjonalitet som er på ønskelisten for en slik app, så oppgaven kan utvides og tilpasses til en viss grad.*
>
> *Et team på to eller tre vil være passende for oppgaven.*
>
> *Prosjektet vil løse en reell oppgave i en av dagens og fremtidens største industrier, og du vil få jobbe med blanke ark, slakke tøyler og moderne teknologi et av Bergens sterkeste utviklermiljøer.*

Studentene ga et godt inntrykk fra første møte, de kom forberedt og hadde gjort seg tanker om fornuftige verktøy å bruke. Da det i mars var tid for at de skulle få kontorplass sammen med oss ble det ganske raskt klart at alt samarbeid, møter og oppfølging måtte foregå online via Microsoft Teams på grunn av Covid-19-situasjonen. Dette har studentene også håndtert på en god måte.

Prosjektet har blitt gjennomført ved ukentlige statusmøter (sprint planning) der en har gått gjennom arbeidet som har vært utført siste uke og diskutert kommende ukes arbeid.
Vizrt har i disse møtene stilt med følgende representanter:

Mikal H. Henriksen - Product Owner
Øyvind Neuman - Senior Software Engineer
Stig Bø - Senior UX Designer
Knut Arvidsson – Senior System Engineer
Roger Sætereng – R&D Manager

Mikal og Øyvind har vært hovedansvarlige for oppfølgingen.

Vi har i tillegg brukt prosjektstyringsverktøyet Jira for å gi studentene en smak på hvordan vi jobber til vanlig.

Vizrt har ikke noe eksisterende retningslinjer for utvikling av mobil apps, så lot vi det være åpent hvilket rammeverk dette prosjektet skulle benytte seg av for å nå målet.

På forhånd hadde vi diskutert ulike alternativer, blant annet:

Fullt native, Java for Android og Swift for iPhone. Her så vi problemer med å vedlikeholde ettersom vi da trengte noen med kompetanse for å rette feil og videreutvikle to ulike produkter.

Xamarin: Ettersom vi Vizrt bruker mye C# så hadde dette kunne vært et naturlig valg, men det kan være litt krevende å komme i gang med.

Ionic Vue, fordi vi bruker Vue for web. Nedsiden med dette er at en ikke har lavnivå tilgang uten å gå tilbake til en native løsning.

Når studentene foreslo **Flutter** så var det noe nytt vi måtte vurdere, men for denne appen så virker det som et meget fornuftig valg. Hvis vi ønsker å gjøre prosessering av bilder og video før opplastning så virker det som om vi kan gjøre det uten å gå veien om fullt native.

Kodegjennomgang viste at det var lett å lese og navigere seg frem i kodebasen uten tidligere erfaring med Flutter spesifikt. Det er nok en kombinasjon av at studentene skrev veldig forståelig kode og at Flutter er bygget opp på en måte som gjør det lett å lese.

Det negative en kan si om Flutter er at det virker være under utvikling og det kan gjøre det vanskelig å henge seg på siste versjon til enhver tid. Akkurat dette er jo ikke et uvanlig fenomen. Vi fikk også kjenne på begrensningene som følger med et rammeverk som prøver å jobbe med minste fellesnevner mellom to plattformer, blant annet med forskjeller i varsler og filutforskere.

Etter at mobilapplikasjonen ble ferdigstilt har vi fått sendt den til én betatester i TV2, og det jobbes med å få noen flere til å teste det.

Vizrt er veldig fornøyd med gjennomføringen av prosjektet på tross av de begrensninger som ble lagt av Covid-19. Studentene har fungert godt sammen, jobbet effektivt og har vært enkle for oss å samarbeide med. De har til tider fått veldig mye tilbakemeldinger om små og store ting i statusmøtene som de har klart å håndtere på en god måte.
Vi sitter igjen med en løsning som har overgått grunnforventningen vår, og vi tror vi med relativt lite arbeid kan bruke det videre ut mot kunder.

Mikal H Henriksen	Øyvind Neuman	Roger Sætereng
Product Owner	Senior Software Engineer	R&D Manager

10.6 Tilbakemelding fra brukertesting

USABILITY TEST ON VIZ STORY UPLOADING APP

Tasks

1. Set up the app by connecting the app to the Viz Story server
2. Add any item from your phone
3. Take and add a picture, then upload it
4. Record and add a short video clip from your phone, then and upload it
5. Preview each of your added items
6. Remove one of the items

Sample

Espen: Videosjef, TV2, Oslo
Mark: Training specialist (former journalist), Vizrt, Bergen
Valentina: UX researcher, Vizrt, Bergen

Findings from three users

1. For one user (Espen), photo and camera don't work (check iPhone model), for the other two only taking photo works.
2. There is no way to close the video preview.
3. Any photo/video taken is not saved on the phone as well.
4. Multi-selections of items from your own gallery are not allowed. Journalists should be able to add more than one item to the list at the time.
5. No way to select multiple items and clear them from the uploading list. So having to clear a list of 10+ items will require a lot of interactions.
6. Uploaded items remain on the list after they have been uploaded. This way they could fill the entire list view and stop the user from seeing additional items
7. In the server name field, the cloud icon has not a clear purpose. When you click it the cursor jumps (almost) back to the start within the text input field.

Please note that access to Story server from an external source shouldn't be given.