Høgskulen på Vestlandet

BACHELOROPPGAVE

«2030 - Sanntidsquiz for Bærekraft»

«Utforsking av vurderinger ved valg av utviklingsmetodikk»

"2030 - Live quiz for Sustainable Development"

"Exploration of assessments for choosing a software development methodology"

Ivar Kvalsund Gjuvsland og Iselin Thorsen Nilsen

Dataingeniør og Informasjonsteknologi

Institutt for datateknologi, elektroteknologi og realfag

Veileder: Adrian Rutle

22/05/2022

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

Høgskulen på Vestlandet

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

TITTELSIDE FOR HOVEDPROSJEKT

Rapportens tittel: «2030 - Sanntidsquiz for Bærekraft» *Rapportens undertittel:* «Utforsking av vurderinger ved valg av utviklingsmetodikk»	*Dato:22/05/2022*
Forfatter(e): Ivar Kvalsund Gjuvsland og Iselin Thorsen Nilsen	*Antall sider u/vedlegg: 56* *Antall sider vedlegg: 60*
Studieretning: Dataingeniør og Informasjonsteknologi	*Antall disketter/CD-er:* 0
Kontaktperson ved studieretning: Adrian Rutle	*Gradering:* Ingen
Merknader:	

Oppdragsgiver: Fana Sparebank	*Oppdragsgivers referanse:*
Oppdragsgivers kontaktperson: (Tidligere Håkon Roald, nå:) Tone Merethe Gamlemshaug	*Telefon:* +47 934 07 332

Sammendrag:

Teamet har utviklet en sanntidsquiz-applikasjon for bærekraft. Denne applikasjonen ble utviklet basert på visjonen og de funksjonelle kravene gitt av Fana Sparebank. Teamet har også utforsket vurderinger gjort ved valg av utviklingsmetodikk mellom to bachelorstudenter ved HVL og én etablert utviklergruppe fra FSB.

Stikkord:

Sanntidsquiz	**Bærekraft**	**Utviklingsmetodikk**

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

Rapporten beskriver arbeidet rundt bachelorprosjektet med tittel «2030 - Sanntidsquiz for Bærekraft», med tilhørende undertittel «Utforsking av vurderinger ved valg av utviklingsmetodikk». Arbeidet fra prosjektet resulterte i en sanntidsquiz-applikasjon, ved navn «2030», samt en utforsking av vurderinger ved valg av utviklingsmetodikk. Prosjektet er gjennomført av Ivar Kvalsund Gjuvsland og Iselin Thorsen Nilsen.

Teamet ønsker å rette en stor takk til:

- Fana Sparebank, oppdragsgiver av prosjektet, for en spennende idé til en mobilapplikasjon og smidighet i forhold til prosjektoppgaven.
- Adrian Rutle, veileder fra HVL, for god veiledning gjennom prosjektets utfordringer og godt samarbeid.
- Håkon Roald, Einar Søreide Johansen og Tone Merethe Gamlemshaug, ansatte fra Fana Sparebank, for godt samarbeid, tillit til teamet og bidrag til prosjektet.
- Medstudenter fra Informasjonsteknologi og Dataingeniør studiene for gode sparring-sesjoner.
- Støttespillere som har bidratt til oppmuntring og motivasjon for prosjektets gjennomføring.

INNHOLDSFORTEGNELSE

FORORD 3

BEGREPSOVERSIKT 1

1 INNLEDNING 3

1.1 KONTEKST 3

1.2 MOTIVASJON 4

1.3 PROSJEKTEIER 4

1.4 MÅL 5

1.5 PROBLEMSTILLING 5

1.6 PROSJEKTETS BEGRENSNINGER 6

1.7 OPPBYGGING AV RAPPORTEN 6

2 PROSJEKTBESKRIVELSE 6

2.1 PRAKTISK BAKGRUNN 6

2.1.1 Tidligere arbeid 6

2.1.2 Initiale krav 7

2.1.3 Initial løsnings-idé 7

2.2 RESSURSER 8

2.3 AVGRENSNINGER 8

2.4 LITTERATUR OM PROBLEMSTILLINGEN 9

2.4.1 Litteratur – Bruk av sanntidsquiz som læremiddel 9

2.4.2 Litteratur – Utforsking av utviklingsmetodikker 9

3 DESIGN AV PROSJEKTET 11

3.1 FORSLAG TIL LØSNING 11

3.1.1 .NET og C# med SignalR 11

3.1.2 Flutter og Dart med Firebase 12

3.1.3 Diskusjon av alternativene 13

3.2 VALGT LØSNING 13

3.3 VALG AV VERKTØY 14

3.4 UTVIKLINGSMETODIKK 14

3.4.1 Utforsking av utviklingsmetodikker 14

3.4.1.1 Waterfall 15
3.4.1.2 Lean 16
3.4.1.3 DevOps 16
3.4.1.4 Scrum 17
3.4.2 Vurderinger ved valg av utviklingsmetodikk 18
3.4.3 Valgt utviklingsmetodikk for prosjektet 18
3.4.4 Prosjektplan 19
3.4.5 Risikovurdering 20
3.5 EVALUERINGSPLAN 20
4 DETALJERT LØSNING 21
4.1 DESIGN AV PRODUKT 21
4.1.1 Visjon for produkt 21
4.1.2 Kravspesifikasjon 21
4.2 UTVIKLINGSPROSESSEN 24
4.2.1 Sprint planlegging 24
4.2.2 Sprint og kodeendringer 24
4.2.3 Retrospekt 27
4.3 SLUTTPRODUKT 28
4.3.1 Programvarearkitektur 28
4.3.2 Klient 29
4.3.3 Firebase 32
5 EVALUERING OG RESULTATER 32
5.1 EVALUERINGSMETODE – PRODUKT 32
5.2 EVALUERINGSRESULTAT - PRODUKT 33
5.3 EVALUERINGSMETODE – VURDERINGER VED UTVIKLINGSMETODIKK 33
5.4 EVALUERINGSRESULTAT – VURDERINGER VED UTVIKLINGSMETODIKK 34
6 DISKUSJON 36
6.1 SLUTTPRODUKTET '2030' 36
6.2 VURDERINGER VED VALG AV UTVIKLINGSMETODIKK 36
6.2.1 Forskjeller ved vurderinger 36

6.2.2 Likheter ved vurderinger *37*

6.2.3 Diskusjon *37*

6.3 UTFORDRINGER 38

6.3.1 Tidsfrist og ansvar *38*

6.3.2 Sammenligning av sluttprodukter *39*

6.3.3 Sluttprodukt *40*

6.4 REFLEKSJONER 40

7 KONKLUSJON OG VIDERE ARBEID 41

7.1 KONKLUSJON 41

7.2 VIDERE ARBEID 42

7.2.1 Relevans av resultater for andre *43*

8 REFERANSER 45

9 VEDLEGG 50

9.1 VEDLEGG 1 – RISIKOANALYSE 50

9.2 VEDLEGG 2 - INTERVJURESULTAT 50

FIGURLISTE

Figur 1 - FNs 17 bærekraftsmål (FN-SAMBANDET, 2022) 3
Figur 2 - Google Scholar søkestreng og søkeresultater 10
Figur 3 – Initial overordnet løsningsarkitektur med .NET og SignalR 11
Figur 4 – Initial overordnet løsningsarkitektur med Flutter og Firebase 12
Figur 5 – Mest brukte utviklingsmetodikker i 2021 (GitLab, 2021) 15
Figur 6 – Waterfall modell livssyklus (Balaji & Murugaiyan, 2012) 15
Figur 7 – Lean utviklingssyklus (Ebert, et al., 2012) 16
Figur 8 – DevOps produksjons- og leveringsprosess (Ebert, et al., 2016) 17
Figur 9 – Scrum prosessen (Abrahamsson, et al., 2002) 17
Figur 10 – GANNT-diagram versjon 2.4 (Prosjekthåndbok) 19
Figur 11 – Brukstilfellediagram for administrator og Quizdeltager 22
Figur 12 – Domenemodell 22
Figur 13 – Wireframes for brukergrensesnitt 23
Figur 14 – Prototype laget i Figma 24
Figur 15 – KanBan-tavle i Azure DevOps 25
Figur 16 – PR i Azure DevOps 26
Figur 17 – Fork Git-klient med branch oversikt 27
Figur 18 – Retrospekt tavle i Azure DevOps 28
Figur 19 – Programvarearkitektur 29
Figur 20 – Startside i ‘2030’ 29
Figur 21 – Klassediagram for de ulike sidene 30
Figur 22 - Brukergrensesnitt for quizdeltager 30
Figur 23 - Brukergrensesnitt for administrator 31
Figur 24 - Klassediagram for klasser med datahåndtering 31
Figur 25 - Databasemodell 32

Begrepsoversikt	
Ord	**Forklaring**
Teamet	Oppdragstaker i prosjektet
FSB	Fana Sparebank (oppdragsgiver)
Utviklergruppe	Fana Sparebank sine systemutviklere
Utviklingsmetodikk	Strukturert metode for å systematisere programvareutvikling
Sprint	Fastsatt tidsperiode for gjennomføring av planlagt utviklingsarbeid
Retrospekt	Evaluering av foregående sprint
Backend	Server siden. Delen av applikasjonen brukerne ikke kan se som kommuniserer med frontend
Frontend	Grafisk brukergrensesnitt for applikasjon. Kommuniserer med backend
Flutter	Rammeverk og SDK for brukergrensesnitt laget av Google
Firebase	Server hosting service
FlutterFire	Firebase programvareutvidelse til Flutter
SDK	Software Development Kit
IDE	Integrated Development Environment – kodeeditor for å utvikle programvare
MVP	Minimum viable product - enkleste fungerende løsning
Avgrensinger	Valg som er tatt (scope på prosjekt, funksjonalitet og tid)
Begrensinger	Ressurser en ikke kan påvirke (teamstørrelse, tidsfrist og kunnskap)
PR	Pull Request - Kodegjennomgang
Branch	Kopi av et kodesegment, enten hele kodebasen eller deler av den, som håndteres i et versjonskontrollsystem
Bugs	Kodefeil i et system, noe som ikke fungerer som tiltenkt eller knekker systemet

Utrulling	Publisering eller oppdatering av kildekode ut i produksjonsmiljø (eller test, staging, osv.)
UI	User Interface - Brukergrensesnitt
Azure DevOps	Utviklingstjeneste, digitalt repositorium
.NET	En utviklingsplattform med åpen kildekode drevet av Microsoft
SignalR	Et kodebibliotek som gjør det enklere å implementere sanntidsfunksjonalitet i .NET applikasjoner
API	Application Programming Interface – grensesnitt for forskjellige program som utveksler informasjon med hverandre

1 INNLEDNING

Bærekraftig utvikling er å ta vare på behovene til mennesker som lever i dag, og samtidig ikke hindre framtidige generasjoners muligheter til å dekke sine. FNs 17 bærekraftsmål er verdens felles arbeidsplan for å utrydde fattigdom, bekjempe ulikhet og stoppe klimaendringene innen 2030, vist i Figur 1 (FN-SAMBANDET, 2022).

Figur 1 - FNs 17 bærekraftsmål (FN-SAMBANDET, 2022)

Bærekraftig utvikling er viktig for alle som ønsker å ta vare på kloden, men til slutt er det barn og unge som skal sitte igjen etter 2030. Derfor er det essensielt at barn og unge lærer om dette slik at de kan kjempe for sine fremtidige muligheter. Sanntidsquiz kan være et engasjerende hjelpemiddel for å løse denne utfordringen.

1.1 Kontekst

Prosjektets produkt «2030 – sanntidsquiz om bærekraft» har opphav i Fana Sparebank (FSB), oppdragsgiver, sitt ønske om å videreutvikle applikasjonskonseptet deres. FSB har et mål om å øke kundeengasjementet hos barn og unge på arrangementer i regi av banken. De har en eksisterende applikasjon - 'Tidi', som har funksjonalitet for primært orienteringsaktiviteter som blir arrangert i Bergen årlig. Det er et ønske fra FSB om å utvide funksjonaliteten til applikasjonen slik at den kan brukes på flere arrangementer.

Denne funksjonaliteten ble utviklet i form av en separat sanntidsquiz-applikasjon. Planen var at det skulle utvikles to versjoner av løsningen, én av FSB sin utviklergruppe og én av teamet. Utviklingen av de to forskjellige versjonene ble isolert fra hverandre i to separate

utviklingsteam, teamet og FSB sin utviklergruppe. I kontekst av dette har prosjektet også handlet om å utforske vurderinger gjort ved valg av utviklingsmetodikk.

Teamet ønsket også å undersøke påvirkningen sanntidsquiz har på engasjement og læring hos barn og unge ved hjelp av applikasjonen. Det ville vært interessant for teamet og FSB å vite om sanntidsquiz er en engasjerende måte å lære om for eksempel bærekraft, i forhold til en presentasjon om det aktuelle temaet. Mer om konteksten til prosjektet og dets visjon finnes i vedlegg «Visjonsdokumentet».

1.2 Motivasjon

I kontekst av produktet, var FSB sin motivasjon for å implementere en sanntidsquiz løsning å bidra til læring blant deltakerne. Ønsket var at det skal være engasjerende og gjøre læring mer fornøyelig. Løsningen som var tiltenkt ville fokusere på å prøve å bidra til læring for barn og unge, med innhold som fokuserer på bærekraft og annen samfunnstematikk.

FSB ønsket ikke å bruke et alternativ som Kahoot! på deres arrangementer, siden de ville sette sitt personlige preg på applikasjonen som skal brukes av kunder. De har også allerede den eksisterende applikasjonen - ‘Tidi’, som de ønsket å utvide med denne funksjonaliteten.

For teamet har både utviklingen av produktet og prosjektet vært en mulighet til å utforske nye teknologier og utviklingsmetodikker. Det har vært interessant å få utforske forskjellige valg av teknologi, og hvordan disse påvirker utviklingen og resultatet av sluttproduktet.

Ved å utforske kompleksiteten av vurderingene som gjøres ved valg av utviklingsmetodikk ønsket teamet å bli bedre rustet til å ta mer overveide valg ved neste utviklingsprosess og få en god innføring i hva et utviklerteam burde legge vekt på.

1.3 Prosjekteier

FSB er en lokalbank i Bergen. De eier merkevarene for både FSB og Himla Banktjenester, samt deres tilhørende eiendomsmegler tjenester. FSB sitt konsept bygger på at de er en tillitsfull lokalbank som skal være «nær der du er». De tar ansvar for og setter søkelys på bærekraft i sitt lokalsamfunn. Derfor er det ønskelig for banken å fremme dette ved hjelp

av produktet som er tiltenkt. Det er et viktig aspekt for FSB å styrke deres konsept og øke kundeengasjement som kan lede til mer fornøyde kunder, og potensielt nye kunder.

Det kunne også vært interessant for FSB å få innsikt i resultatene av prosjektet, som gjelder undersøkelsen av påvirkningen sanntidsquiz har på barn og unge, samt utforskingen av vurderinger gjort ved valg av utviklingsmetodikk. Dette kan brukes av FSB for å få mer innsikt i barn og unge som kundegruppe og kan muligens forbedre deres rutiner og metoder for programvareutvikling.

Tone Merethe Gamlemshaug representerer FSB som prosjekteier.

1.4 Mål

Ett av målene til teamet var å utvikle en quizapplikasjon med sanntidsfunksjonalitet. Applikasjonen skulle utvikles som en separat løsning, ved navn '2030'. Sanntidsquiz-applikasjonen skulle innfri ønsket om å bidra til læring blant barn og unge om samfunnstematikk som bærekraft og økonomi.

Et annet mål var å se på forskjellene i vurderinger gjort ved valg av utviklingsmetodikk mellom to bachelorstudenter ved HVL og én etablert utviklergruppe fra FSB. Ved å få muligheten til å utforske dette var også et undermål at teamet skulle bli bedre forberedt på å gjøre dette igjen senere i arbeidslivet.

1.5 Problemstilling

Basert på konteksten, motivasjonen og målene har teamet kommet frem til to problemstillinger med tilhørende forskningsspørsmål:

1. Hvordan kan teamet utvikle en sanntidsquiz-applikasjon som bidrar til læring om bærekraft blant barn og unge?
    - *Hvordan påvirkes 4. klassingers resultat og engasjement i læring om bærekraft ved bruk av sanntidsquiz fremfor presentasjon?*

2. Hvordan kan teamet utforske vurderinger som er gjort av to bachelorstudenter i forhold til én etablert utviklergruppe ved valg av utviklingsmetodikk?

- *Hvilke vurderinger gjøres annerledes av to bachelorstudenter ved HVL i forhold til én etablert utviklergruppe fra FSB når det gjelder valg av utviklingsmetodikk for ett prosjekt?*

1.6 Prosjektets begrensninger

Prosjektet har hatt en del større endringer etter prosjektstart som har påvirket både prosjektets fokusområder, metoder for gjennomføring og resultatene av gjennomføringen. Prosjektoppgaven som teamet ble presentert og ble enige med FSB om før oppstarten av prosjektet var ganske annerledes fra det teamet har endt opp med. Rapportens innhold og struktur vil derfor reflektere dette og vike til dels fra den oppsatte malen.

På grunnlag av disse endringene, som diskuteres i detalj i kapittel 6.3, ble prosjektet forsinket etter prosjektstart, spesielt i forhold til utviklingen av produktet.

1.7 Oppbygging av rapporten

Det neste kapittelet tar for seg beskrivelsen av prosjektet. I kapittel 3 presenteres løsningsidéer, valgt løsning, utforsking av utviklingsmetodikker og prosjektplanen. Kapittel 4 beskriver dagens løsning og sluttprodukt i detalj, samt prosessen for å utvikle den. Evalueringer og resultater presenteres i kapittel 5 og diskuteres videre i kapittel 6. I dette kapittelet diskuteres også prosjektets utfordringer i detalj. Rapporten avslutter med konklusjonene som er nådd i kapittel 7, samt videre arbeid og relevans av resultatene for andre.

2 PROSJEKTBESKRIVELSE

2.1 Praktisk bakgrunn

Bakgrunnen for prosjektet og dets opphav baserer seg på det tidligere arbeidet, de initiale kravene og den initiale løsnings-idéen.

2.1.1 Tidligere arbeid

Utgangspunktet for applikasjonen var arbeidet som er gjort tidligere av FSB i ‘Tidi’. Arbeidet består av et API for «aktiviteter» i applikasjonen og en generell frontend for å

vise frem informasjonen fra tilhørende API. Utvidelsen av denne applikasjonen ble utviklet som et separat produkt og skulle senere implementeres som en egen aktivitet i 'Tidi', som inneholder logikk for en sanntidsquiz.

Quiz i seg selv er ikke et nytt konsept og informasjon om hva en quiz skal inneholde er standardisert i form av spørsmål og tilhørende riktige og feil svaralternativer. Det finnes allerede flere løsninger for å kunne gjennomføre en sanntidsquiz på et arrangement. De største konseptuelle inspirasjonskildene for denne applikasjonen er Kahoot (Kahoot! AS, 2022) og NRKs "Alle mot 1".

2.1.2 Initiale krav

De initiale overordnede kravene fra FSB ble gitt i form av den originale oppgavebeskrivelsen. Beskrivelsen baserte seg på å utvide den eksisterende 'Tidi'-applikasjonen med ny funksjonalitet i form av en sanntidsquiz som kan benyttes i forbindelse med arrangementer i regi av FSB. Det ble også stilt krav til hvilken teknologi teamet skulle bruke for å utvikle applikasjonen.

Grunnet endringene i prosjektet, forklart nærmere i kapittel 6.3, skiftet kravene til at produktet skulle være en selvstendig sanntidsquiz, og det skulle lages to versjoner av den, én av teamet og én av FBS.

Applikasjonen skal ha en administrator som styrer gangen i quizen. Alle som har '2030' applikasjonen lastet ned på sin mobile enhet og internettforbindelse skal kunne delta enkelt ved å angi et brukernavn og trykke delta. Når quizdeltager får opp et spørsmål skal applikasjonen vise hvor lang tid en har på å svare, vise hva quizdeltageren har svart og hva det korrekte svaret var når tiden har gått ut. Brukeren som er administrator skal ha mulighet til å starte, styre, redigere, slette og opprette en ny quiz. Disse kravene er beskrevet nærmere i visjonsdokumentet og i vedlegg «Kravdokumentasjon».

2.1.3 Initial løsnings-idé

Oppgavebeskrivelsen opprinnelig speilet av FSB baserte seg på at løsningen skulle bestå av flere komponenter. Én backend som håndterer quizflyt og funksjonalitet for quizen, og én eller flere frontend som visualiserer denne flyten. Det var et API som skulle inneholde datastrukturen og logikken for endring av quiz informasjonen. Frontend skulle ta seg av alt

som skal vises til brukeren. Her blir data fra backend strukturert og presentert for brukeren. Det skulle også være en sanntidskomponent som blir flettet sammen med de andre komponentene slik at applikasjonen blir oppdatert i sanntid for alle brukere. Disse komponentene kan lages med flere teknologier, men siden det opprinnelige kravet var å bruke .NET plattformen og C# programmeringsspråket, sammen med et sanntidsbibliotek, SignalR, baserte den initiale løsningsidéen seg på dette.

2.2 Ressurser

FSB tilbydde både kontorplass, laptop og nødvendig software. De satt opp eget prosjekt til teamet, med et tomt digitalt repositorium, i utviklingstjenesten Azure DevOps. Her kunne teamet utvikle kodebasen sin fra grunnen av. I prosjektet var det en Kanban-tavle som teamet brukte for å ha kontroll på oppgavelisten, aktive oppgaver og hvem som jobbet med hva. Azure DevOps gav også teamet mulighet til å gjennomføre en PR, kodegjennomgang, for hver kodeendring i kodebasen.

FSB er også en ressurs i form av kunnskap og erfaring med utviklingsprosesser og teknologi. De kan blant annet bistå med teknisk hjelp hvis det skulle være noe problemer med bruk av Azure DevOps, andre tekniske verktøy eller maskinvaren. Teamet har likevel ikke brukt de som ressurs da teamet ikke ønsket å bli påvirket av måten de valgte å legge opp utviklingsløpet sitt.

2.3 Avgrensninger

De funksjonelle kravene fra FSB var konkrete og veldefinerte og trengte derfor ingen avgrensinger slik teamet så det.

På grunn av utfordringene diskutert i kapittel 6.3 endte teamet opp med å avgrense produktet i oppgaven til å være en enkleste fungerende løsning (MVP) med kun de viktigste funksjonalitetene demonstrert.

Evalueringsmetoden for utforskingen av vurderinger rundt valg av utviklingsmetodikk måtte også avgrenses fra et muntlig intervju med åpne spørsmål til et skriftlig semistrukturert intervju.

2.4 Litteratur om problemstillingen

Teamet har utforsket relevant litteratur som forberedelse til løsninger av problemstillingene. Litteraturen omhandler sanntidsquiz som læremiddel og utforsking av forskjellige utviklingsmetodikker.

2.4.1 Litteratur – Bruk av sanntidsquiz som læremiddel

Studenter har et positivt inntrykk ved bruk av Kahoot! som læremiddel (Wang & Tahir, 2020). Dette resulterte i en positiv effekt på motivasjon, engasjement, konsentrasjon, oppfattet læringsutbytte, oppmerksomhet og fornøyelighet (2020).

Med grunnlag i dette tenkte teamet at sanntidsquiz kan være et godt egnet verktøy for korte læringsøkter hvor en ønsker større engasjement innenfor et aktuelt tema. Spillifisering av læring kan bli brukt i de situasjonene hvor det er ønskelig å gjøre noe ekstra ut av det aktuelle temaet som deltagerne skal lære noe om. Dette er direkte overførbart til teamet sin applikasjon og en besvarelse av forskningsspørsmålet rundt engasjement og læring ved bruk av sanntidsquiz.

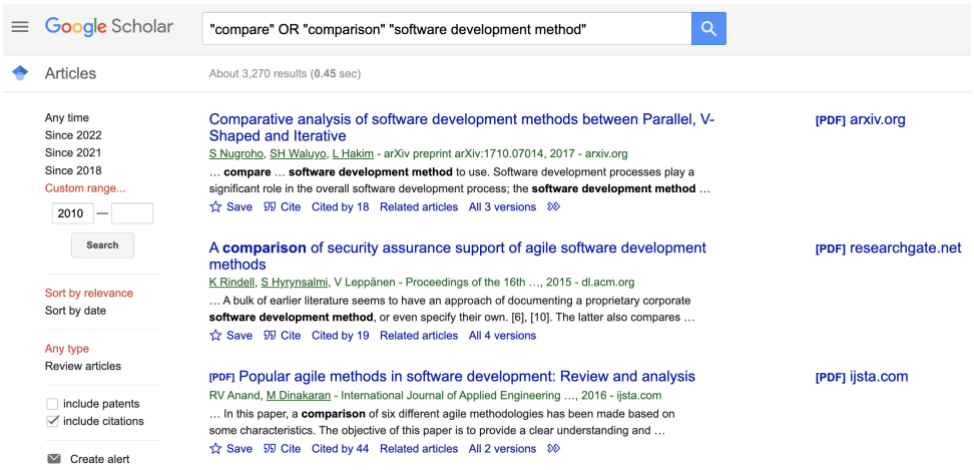
2.4.2 Litteratur – Utforsking av utviklingsmetodikker

Teamet gjorde et litteratursøk for å utforske vurderinger ved valg av utviklingsmetodikk og for å ta et veloverveid valg.

For å finne litteratur om utforsking og sammenligning av utviklingsmetodikker søkte teamet i Google Scholar med følgende søkestreng: «"compare" OR "comparison" "software development method"», sammen med filtrering på resultater fra 2010 og oppover, vist i Figur 2. Med denne søkestrengen fikk teamet totalt 3270 søkeresultater, men så kun på litteratur som oppfylte følgende kriterier:

- Så nye som mulig – programvareutvikling er et fagfelt som utvikler seg hyppig, derfor er det viktig å følge med på for eksempel økende popularitet av nye metodikker
- Inneholdt direkte sammenligning av utviklingsmetodikker – ikke deler innenfor dette
- Faglig artikkel – rapport eller artikkel i tidsskrift

- Tilgjengelig litteratur – ikke all litteratur som passet de ovennevnte kravene var gratis tilgjengelig for teamet og ble dermed ikke mulig å lese eller bruke



Figur 2 - Google Scholar søkestreng og søkeresultater

Systemutviklingsmetoder brukes for å utvikle programvare på en systematisk måte, med mål om å øke sannsynligheten for at programvaren er ferdig utviklet innen tidsfristen og er av høy kvalitet (Mishra & Dubey, 2013). Det finnes en rekke utviklingsmetodikker som hjelper utviklere i utviklingsprosesser (2013). For hvert utviklerteam, og i noen tilfeller for hvert prosjekt, er det nødvendig å gjøre ett sett med vurderinger for å kunne velge ut den metodikken som passer best. Det er ingen fasit på hvilken metodikk som er mest effektiv eller best i praksis, men metodikken som velges må passe den individuelle utvikleren, det individuelle utviklerteamet og produktet som skal utvikles (Nugroho, et al., 2017).

For at teamet kunne utforske vurderingene som blir gjort ved valg av utviklingsmetodikk mellom to studenter ved HVL og én utviklergruppe fra FSB, var det nødvendig å se på hvilke kjennetegn som inngår i en tradisjonell utviklingsmetodikk. Kjennetegnene er blant annet; klare krav til systemet, nøye planlegging, veldefinerte problemstillinger, at prosessene er forutsigbare og kan settes innenfor passende tidsrammer, og bli kontrollert gjennom utviklingen (Dora & Dubey, 2013). Det var meningen at vurderingene skulle kommuniseres av FSB, uten særlige innspill fra teamet. Disse kjennetegnene var derfor bare en inspirasjonskilde, siden teamet la også andre personlige vurderinger til grunn ved valg av utviklingsmetodikk.

3 DESIGN AV PROSJEKTET

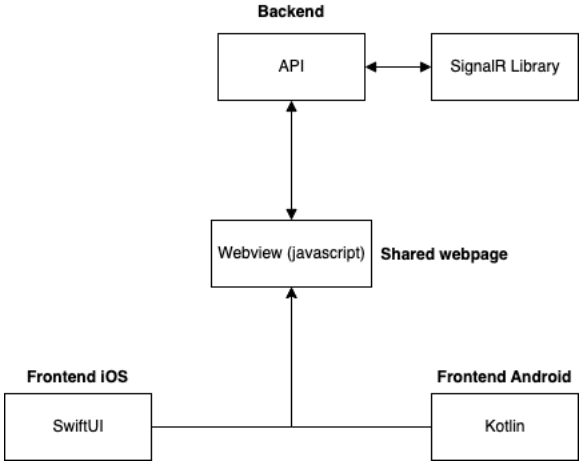
3.1 Forslag til løsning

I planleggingsfasen av prosjektet har teamet brukt mye tid på å diskutere og undersøke forskjellige teknologier som kan fungere for å utvikle applikasjonen. Teamet valgte å undersøke nærmere .NET med SignalR og Flutter rammeverket med Firebase server-tjenester.

3.1.1 .NET og C# med SignalR

.NET og C# er FSB sin etablerte standard for utvikling. SignalR, som kan brukes sammen med .NET, tilbyr implementasjon av sanntidsfunksjonalitet som kan brukes i quizapplikasjonen (Microsoft, 2022). Med denne løsningen må teamet selv sette opp et eget API for dataflyt mellom frontend og backend. Da må teamet også sette opp to forskjellige frontend, ett for iOS og ett for Android. Dette gjøres ved å skrive i programmeringsspråket Swift for iOS og programmeringsspråket Kotlin for Android.

Tiltenkt løsningsarkitektur er å lage et Webview i Javascript, for at frontendene på iOS og Android skal kunne kommunisere med API-et. Dette er et felles nettsted, et mellomledd, som kommuniserer med API-et og de to separate frontendene, som illustrert i Figur 3. Bruk av SignalR er ukjent for både teamet og FSB.



Figur 3 – Initial overordnet løsningsarkitektur med .NET og SignalR

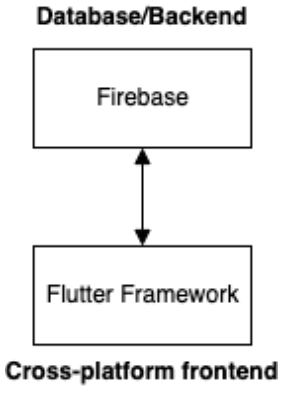
En av fordelene med å bruke .NET og C# er at teamet er kjent med dette fra før. Det var også mulig for teamet å få hjelp av FSB til det praktiske rundt applikasjonen og noe av utviklingen.

En av ulempene med denne løsningen er at visse komponenter av SignalR ikke støtter både Swift og Kotlin (Microsoft, 2022).

3.1.2 Flutter og Dart med Firebase

Flutter er Google sitt rammeverk for å blant annet bygge multiplattform mobilapplikasjoner (Google, u.d.). Flutter sammen med Firebase tilbyr sanntid database implementasjon, som er essensielt for applikasjonen. Firebase kan gjøre det lettere for teamet å få med dette, siden de tar seg av mye av implementasjonen (Google, 2022). Dart er programmeringsspråket som Flutter baserer seg på. Bruk av Flutter og Dart med Firebase er i utgangspunktet gratis så lenge en har en Google konto. Hvis en ønsker utvidet funksjonalitet av Firebase, koster dette en månedlig sum som varierer basert på funksjonalitet.

Flutter muliggjør å utvikle én versjon av applikasjonen for både iOS og Android, altså én felles kodebase. Applikasjonen kommuniserer direkte med Firebase, illustrert i Figur 4. Dette kan gjøre omfanget av utviklingsprosessen mindre og teamet kan derfor legge mer energi i å lage en applikasjon av høyere funksjonell kvalitet.



Figur 4 – Initial overordnet løsningsarkitektur med Flutter og Firebase

Det er flere fordeler med Flutter og Firebase; felles kodebase, maler for komponenter som teamet kan ta inspirasjon fra, og den viktigste, at Firebase tilbyr mye av funksjonaliteten

som er nødvendig for applikasjonen. Dette gjelder blant annet sanntid database håndtering og API.

Noen ulemper med Flutter og Firebase er at det var helt nytt for teamet, og det er ingen utviklere i FSB som har jobbet med dette før. Dart som brukes for Flutter er også nytt. Valg av ukjente teknologier påvirker tiden utviklingen tar, da det må tilrettelegges for opplæring.

3.1.3 Diskusjon av alternativene

Under utforskingen av de ovennevnte løsningene ble det klart for teamet at det er visse kriterier som burde prioriteres over andre. Blant annet var det viktig at fokuset kunne ligge på utviklingen av funksjonaliteten på grunn av tidspress. Det burde derfor legges vekt på teknologier som tilbyr en form for multiplattform utviklingsløsning.

Det er sterke fordeler med begge alternativene. Med flere utviklere og ressurser, ville det nok gitt mening å ta i bruk .NET med SignalR grunnet fordelene som følger med det valget. Likevel er det faktorer teamet tok hensyn til som ikke sammenfalt helt med å velge .NET. Det er begrenset med tid satt av til utviklingen, og siden en fungerende applikasjon vektlegges er det viktig for teamet å begrense omfanget av utviklingsprosessen med å velge en multiplattform utviklingsløsning og bruke ferdig funksjonalitet fra Firebase.

Til tross for liten erfaring valgte teamet likevel løsningen med Flutter og Dart med Firebase basert på fordelene. FSB valgte .NET med SignalR, som bygger på deres etablerte standard.

Det er flere grunner til at teamet endte opp med å velge Flutter og Dart med Firebase, men dette utdypes i kapittel 6.3.

3.2 Valgt løsning

Flutter og Dart med Firebase er teknologiene teamet bestemte seg for å bruke, med grunnlag i utforskingen og diskusjonen ovenfor. Det skal ved hjelp av disse teknologiene lages en sanntidsquiz-applikasjon. Det vil da være en felles frontend med felles kodebase for begge plattformer, iOS og Android, som snakker direkte med sanntidsdatabasen i

Firebase. Kodebasen vil skrives i programmeringsspråket Dart som samarbeider med rammeverket, Flutter.

3.3 Valg av verktøy

Nødvendige verktøy for utvikling med Flutter og Firebase:

- Flutter SDK (installert på lokal maskin)
- Android Studio IDE (til simulering)
- Xcode IDE (til simulering)
- Visual Studio Code IDE
- Fork (Git klient)
- Git (system for versjonskontroll installert på lokal maskin)
- Azure DevOps (oppbevaring av kildekode, mfl.)

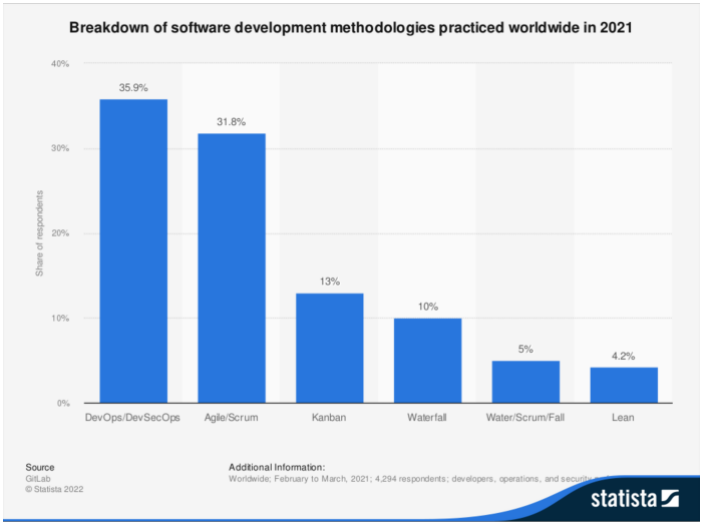
3.4 Utviklingsmetodikk

En utviklingsmetodikk er nyttig for at et utviklerteam skal ha en felles forståelse for hvordan utviklingen av et produkt skal gjennomføres.

For å kunne ta et veloverveid valg av utviklingsmetodikk måtte teamet utforske flere alternativer. Det var essensielt at teamet hadde satt seg godt inn i forskjellige karakteristikker ved forskjellige utviklingsmetodikker. Teamet undersøkte også vurderinger et utviklerteam gjør for å kunne skille mellom metodikker og velge den mest passende. Dette legger grunnlaget for utforskingen av utviklingsmetodikker. Utforskingen var viktig for prosjektets omfang og fokusområde, slik at teamet kunne løse problemstillingen; «Hvordan kan teamet utforske vurderinger som er gjort av to bachelorstudenter i forhold til en etablert utviklergruppe ved valg av utviklingsmetodikk?»

3.4.1 Utforsking av utviklingsmetodikker

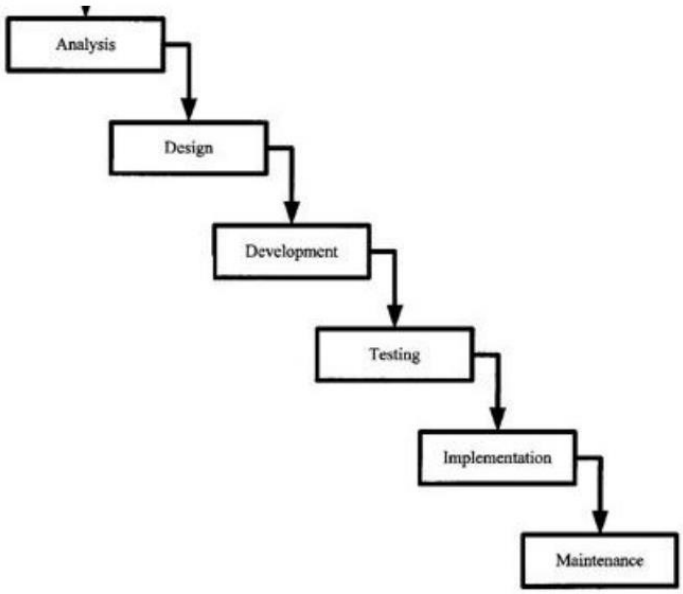
Teamet valgte å sette seg inn i Waterfall, Scrum, Lean og DevOps fordi dette er noen av de mest brukte utviklingsmetodikkene i dag, vist i Figur 5 (GitLab, 2021).



Figur 5 – Mest brukte utviklingsmetodikker i 2021 (GitLab, 2021)

3.4.1.1 Waterfall

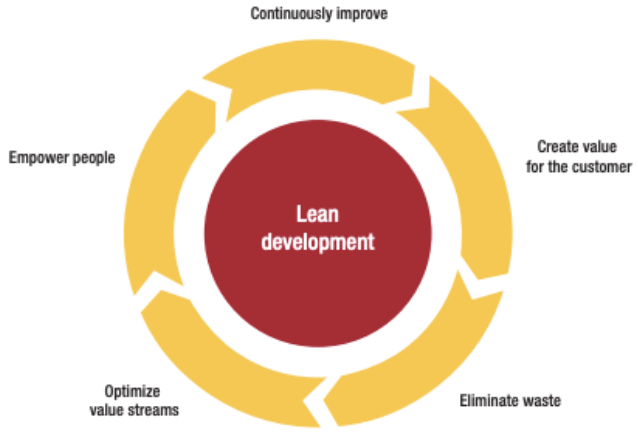
Waterfall er en utviklingsmetodikk som er lagt opp som en sekvensiell utviklingsmodell, illustrert i Figur 6 (Balaji & Murugaiyan, 2012). Utviklingsprosessen baserer seg på å gjennomføre en fase før en går videre på neste. Hver fase bli fullført innenfor en gitt tidsramme og uten overlapp med andre faser. Det er en rigid modell som krever at utviklerteamet etterstreber å fullføre design fasen fullstendig før en kan gå videre med utviklingsarbeidet (Balaji & Murugaiyan, 2012).



Figur 6 – Waterfall modell livssyklus (Balaji & Murugaiyan, 2012)

3.4.1.2 Lean

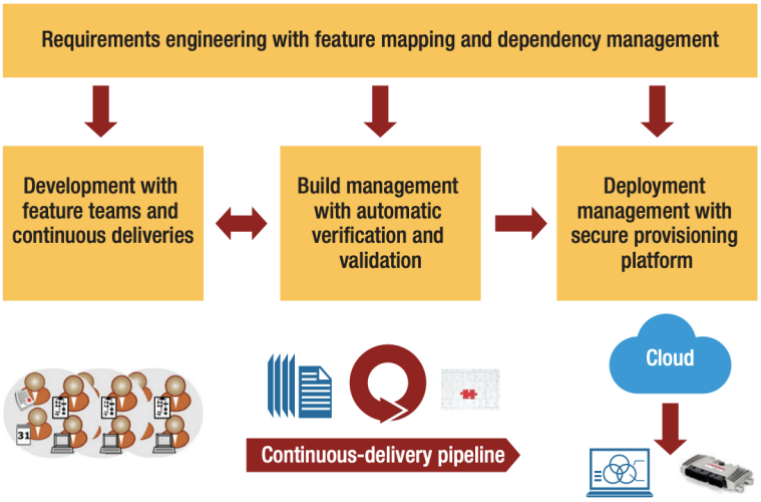
Lean bygger på å lage en så enkel løsning som mulig, levere dette som et produkt også inkrementelt bygge videre på produktet (Pranczke, 2021). Metodikken fokuserer på å skape verdi for kunden, eliminere sløsing, optimalisere verdistrømmer, styrke mennesker og kontinuerlig forbedre seg (Ebert, et al., 2012). Lean starter med verdiorientering, for så å bevege seg videre inn i de andre fokusområdene, illustrert i Figur 7. Lean fokuserer heller ikke spesielt mye på hvordan en skal utvikle programvare, men fungerer mer som en prosjektmetodikk som kan tilpasses mange forskjellige bransjer. Det er likevel visse prinsipper som blir brukt av Lean metodikken i programvareutvikling uten at et team nødvendigvis implementerer hele metodikken og dens filosofi (Ebert, et al., 2012).



Figur 7 – Lean utviklingssyklus (Ebert, et al., 2012)

3.4.1.3 DevOps

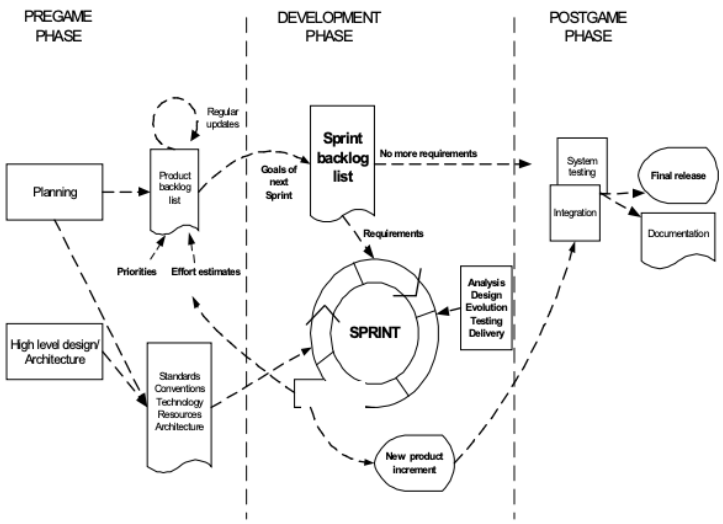
DevOps handler om rask og fleksibel utvikling (Ebert, et al., 2016). Metodikken integrer effektivt; utvikling, levering og drift. På denne måten fasiliterer metodikken en smidig og flytende forbindelse mellom disse delene som vanligvis er separerte, illustrert i Figur 8 (Ebert, et al., 2016). Denne metodikken fungerer derfor bra for et utviklerteam som må fokusere på å utvikle nye systemer og videreutvikle de allerede eksisterende systemene. DevOps har stort fokus på kontinuerlig integrasjon og levering. Utvikling og drift kan kombineres ved hjelp av kryssfunksjonelle team som jobber med kontinuerlige leveranser av operative funksjoner (Ebert, et al., 2016).



Figur 8 – DevOps produksjons- og leveringsprosess (Ebert, et al., 2016)

3.4.1.4 Scrum

Scrum er en empirisk måte å implementere idéene fra en industriell prosess i en systemutviklingsmetode. Metodikken baserer seg på fleksibilitet, tilpasning og produktivitet (Abrahamsson, et al., 2002). Scrum fokuserer ikke på en spesifikk måte å implementere programvaren, men heller hvordan et team skal fungere for å kunne tilpasse et system i et stadig endrende miljø (Abrahamsson, et al., 2002). Idéen med Scrum er at systemutvikling har flere bevegelige deler som har stor sannsynlighet for endring gjennom løpet, som betyr at prosessen må være fleksibel nok til å kunne ta høyde for dette. Bruk av sprinter med retrospekt og sprintplanlegging tilrettelegger for dette. Scrum sin prosess illustreres i Figur 9.



Figur 9 – Scrum prosessen (Abrahamsson, et al., 2002)

3.4.2 Vurderinger ved valg av utviklingsmetodikk

For å kunne velge riktig utviklingsmetodikk for et prosjekt må et utviklerteam gjøre en rekke vurderinger. På forhånd av prosjektets start har teamet innad diskutert hvilke vurderinger som er relevante for prosjektet og hva som må prioriteres:

- Teamets størrelse – 2 utviklere som fungerer som blant annet prosjektledere, designere og testere.
- Teamets kompetanse – nesten nyutdannede junior utviklere
    - Det må avsettes tid til opplæring av ny teknologi
- Tydelige krav fra oppdragsgiver fra begynnelsen av – det kom ikke til å bli noen endringer av betydning når det kom til funksjonalitetskrav til produktet
- Produktets minimum krav til funksjonalitet (MVP) – ikke mulighet eller tid til for mange eller for store iterasjoner med revurdering av funksjonalitet
- Smidighet – metodikken må tillate at uforutsette hendelser kan skje og at ting kan bli forskjøvet og forsinket
- Bruk av kodekontroll – PR
- Retrospekt etter hver sprint
- Korte sprinter med klare krav og oppgaveplanlegging

3.4.3 Valgt utviklingsmetodikk for prosjektet

Teamet landet på at en tilnærming av Scrum ville passe best for prosjektet. Hovedgrunnen til at Scrum ble valgt var fordi metodikken bygger på blant annet at teamet er transparent og at utviklingen og produktet hyppig tilpasses (Schwaber, 2020).

Det er ikke mulig å følge Scrum som utviklingsmetodikk fullstendig med et team som kun består av to utviklere, men teamet har valgt å ta utgangspunkt i følgende deler av Scrum teorien.

Teamets tilnærming av Scrum består overordnet av:

- Utviklingsperiode på 5 uker
- Ukentlig sprint
    - Sprint planlegging – begynnelsen av hver sprint

        - Hvorfor har vi denne sprinten?
        - Sprintmål?
        - Hva kan bli gjort?
        - Hvordan skal det bli gjort?
- Retrospekt etter hver sprint
    - Hva var bra?
    - Hva var dårlig?
    - Hva kan forbedres i neste sprint?
- Artefakter – kontrollert ved hjelp av Kanban tavle
    - Produkt backlog – Produktmål og alle oppgaver
    - Sprint backlog – Sprintmål og tilhørende oppgaver
    - Inkrement – CLOSED work items (når PR er fullført)

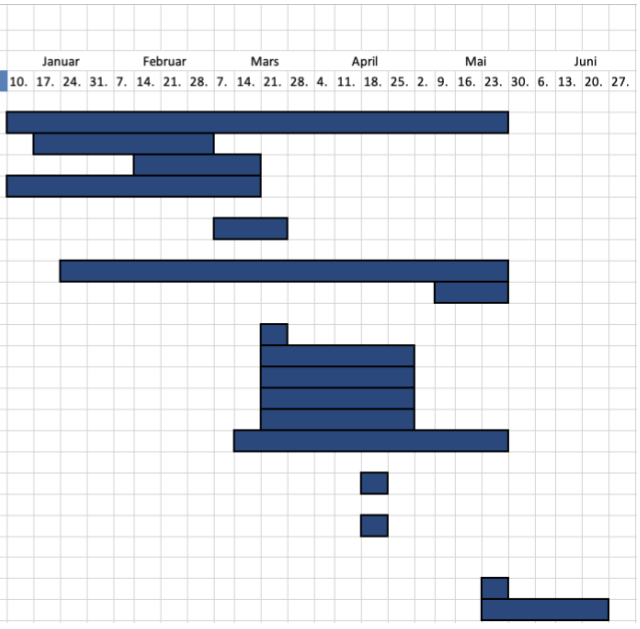
3.4.4 Prosjektplan

Prosjektplanen er utviklet i form av et GANTT-diagram, vist i Figur 10. I GANTT-diagrammet blir de forskjellige fasene av prosjektet, med definerte underaktiviteter, visuelt representert med tilhørende detaljer. GANTT-diagrammet har blitt inkrementelt revidert gjennom prosjektet.

Utforsking vurderinger: utviklingsmetodikk — Prosjekt start: 10/01-2022

Ivar Kvalsund Gjuvsland og Iselin Thorsen Nilsen — Gruppe D2

Identifikator	Ansvarsperson	Status	Oppstart	Slutt	Estimert timeantall
Fase 1 - Planlegging og oppstart					
Prosjekthåndbok	Iselin	Ferdig	11/01/2022	23/05/2022	60 timer
Visjonsdokument	Ivar	Ferdig	18/01/2022	25/02/2022	30 timer
Forprosjektrapport	Ivar og Iselin	Ferdig	15/02/2022	11/03/2022	30 timer
Prosjektplanlegging	Ivar og Iselin	Ferdig	11/01/2022	11/03/2022	20 timer
Fase 2 - Utvikling og skriving					
Opplæring ny teknologi	Ivar og Iselin	Ferdig			
Dokumentasjon					90 timer
Kravdokumentasjon	Ivar og Iselin	Ferdig	22/02/2022	23/05/2022	60 timer
Systemdokumentasjon	Ivar og Iselin	Ferdig	25/04/2022	23/05/2022	30 timer
Utvikling					160 timer
Oppsett	Ivar og Iselin	Ferdig	22/03/2022	27/03/2022	10 timer
Bruker	Ivar og Iselin	Påbegynt	28/03/2022	25/04/2022	30 timer
Administrator	Ivar og Iselin	Ferdig	18/04/2022	25/04/2022	40 timer
Database og sanntid	Ivar	Ferdig	04/04/2022	25/04/2022	50 timer
Quiz	Iselin	Ferdig	28/03/2022	25/04/2022	20 timer
Hovedrapport	Ivar og Iselin	Ferdig	15/03/2022	23/05/2022	120 timer
Fase 3 - Evaluering og sammenligning					
Planlegging av sammenligning og evaluering	Ivar og Iselin	Ferdig	18/04/2022	20/04/2022	10 timer
Evaluering av teamet	Ivar og Iselin	Ikke ferdig	Ikke aktuelt	Ikke aktuelt	10 timer
Evaluering av FSB	Ivar og Iselin	Ferdig	22/04/2022	22/04/2022	10 timer
Sammenligning av teamet og FSB	Ivar og Iselin	Ikke ferdig	Ikke aktuelt	Ikke aktuelt	40 timer
Fase 4 - Avsluttende					
Refleksjonsnotat	Ivar og Iselin	Ikke ferdig	24/05/2022	26/05/2022	5 timer
Expo	Ivar og Iselin	Ikke ferdig	24/05/2022	15/06/2022	20 timer



Figur 10 – GANNT-diagram versjon 2.4 (Prosjekthåndbok)

3.4.5 Risikovurdering

Teamet har gjennomført en risikovurdering, som er vedlagt i Vedlegg 1 – Risikoanalyse og i vedlegg «Prosjekthåndbok», i forkant av prosjektets oppstart som fungerte som en guide for mulige risikoer i prosjektet. Denne har også blitt revidert i løpet av prosjektet. De viktigste risikomomentene fra denne analysen er misbruk av personinformasjon, sammenligning av produkter blir vanskelig å gjennomføre, produktene som skal sammenlignes er for like og for lite å sammenligne. Sistnevnte utspilte seg og diskuteres i kapittel 6.3.

3.5 Evalueringsplan

Prosjektet sin overordnede evalueringsplan baserte seg på å prøve å besvare forskningsspørsmålene. For evalueringen av produktet ville teamet forsøke å gjøre dette med en form av brukertesting i en 4. klasse og resultatene fra dette skulle analyseres. Det var ønskelig å gjennomføre evalueringen slik fordi teamet ville fokusere på å få tilbakemeldinger fra et segment fra målgruppen. Slik kunne teamet få relevant innsikt i effekten applikasjonen faktisk hadde på læring og engasjement hos barn og unge. Brukertesting er også viktig for å avdekke eventuelle bugs i applikasjonen, så dette kunne teamet også fått tilbakemeldinger på.

Når det kom til å evaluere vurderinger gjort ved valg av utviklingsmetodikk skulle teamet gjennomføre et muntlig semistrukturert intervju (Andersen, 2020). Dette skulle gjennomføres med ett intervjuobjekt; én systemutvikler fra FSB. Teamet skulle da utforske forskjellene og likhetene mellom teamet og utviklergruppen fra FSB sine vurderinger. Teamet valgte å evaluere på denne måten for å få konkrete svar fra FSB på det teamet lurte på. For å avdekke disse vurderingene måtte teamet planlegge hvilke spørsmål som skulle stilles og hvordan teamet kunne få åpne svar som fortsatt inneholdt den relevante informasjonen, men med potensielt ytterligere interessante funn.

Denne og ovennevnt metode beskrives nærmere i kapittel 5.

4 DETALJERT LØSNING

I denne delen blir prosessen og produktet i form av applikasjonen, ‘2030’ som er utviklet presentert i detalj. Konkret blir designet av produktet, gjennomføringen av utviklingsperioden samt sluttproduktet forklart.

4.1 Design av produkt

Designet av produktet er utformet ved en iterativ prosess mellom teamet, FSB og veileder. Gjennom denne prosessen er det blitt enighet om visjon for produktet, krav og funksjonalitet. Teamet har selvstendig utformet kravdokumentasjonen ut fra de spesifikasjonene.

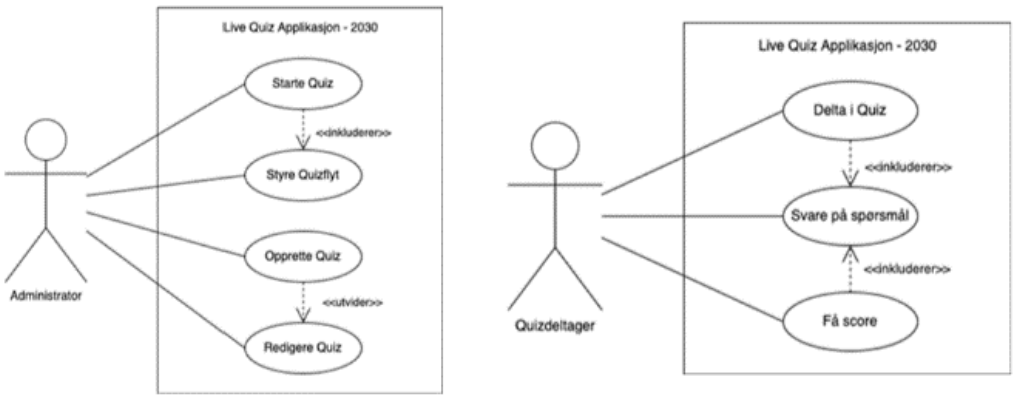
4.1.1 Visjon for produkt

Prosessen med å utforme visjonsdokumentet tok utgangspunkt i prosjektoppgaven gitt av FSB. Visjonen for produktet skulle bygge en felles forståelse mellom FSB, teamet og veileder. Teamet fikk kartlagt hvem som er brukere av produktet, hvilket problem produktet skulle løse og hva FSB ønsket av funksjonalitet. Dette er utdypet i vedlegg «Visjonsdokument».

De funksjonelle kravene fra visjonsdokumentet la grunnlaget for utformingen av kravspesifikasjonen.

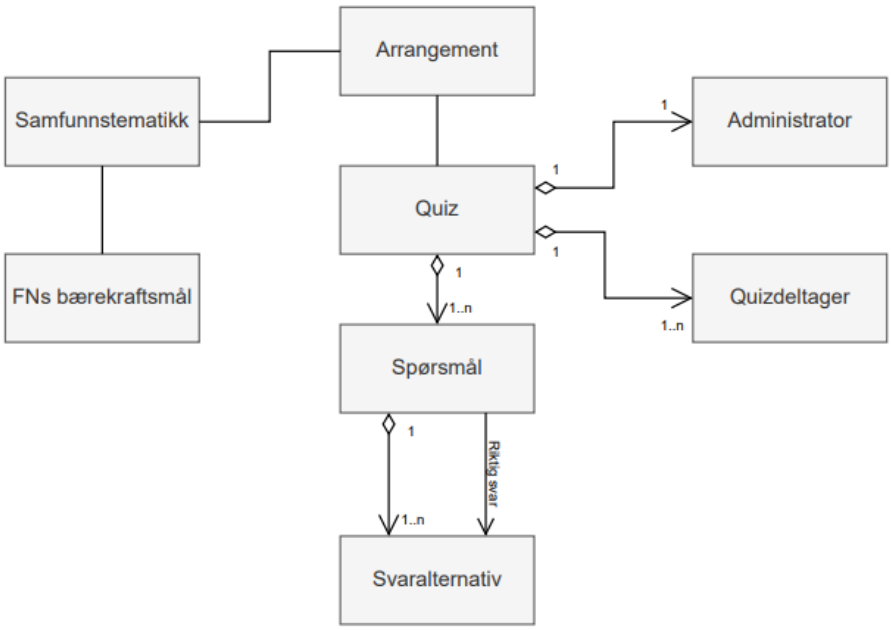
4.1.2 Kravspesifikasjon

Kravspesifikasjonen fastsetter ønskede funksjonaliteter, utdypet i vedlegg «Kravdokumentasjon». Der defineres brukerdesign og formaliseringen av kravene til produktet. Funksjonalitetene er beskrevet gjennom brukstilfellediagram og brukerreiser. Figur 11 viser funksjonaliteten som administrator og quizdeltager kan benytte seg av. I hovedsak dreier det seg om sentrale elementer som må med for at en sanntidsquiz kan gjennomføres med en lignende flyt som i Kahoot!.



Figur 11 – Brukstilfellediagram for administrator og Quizdeltager

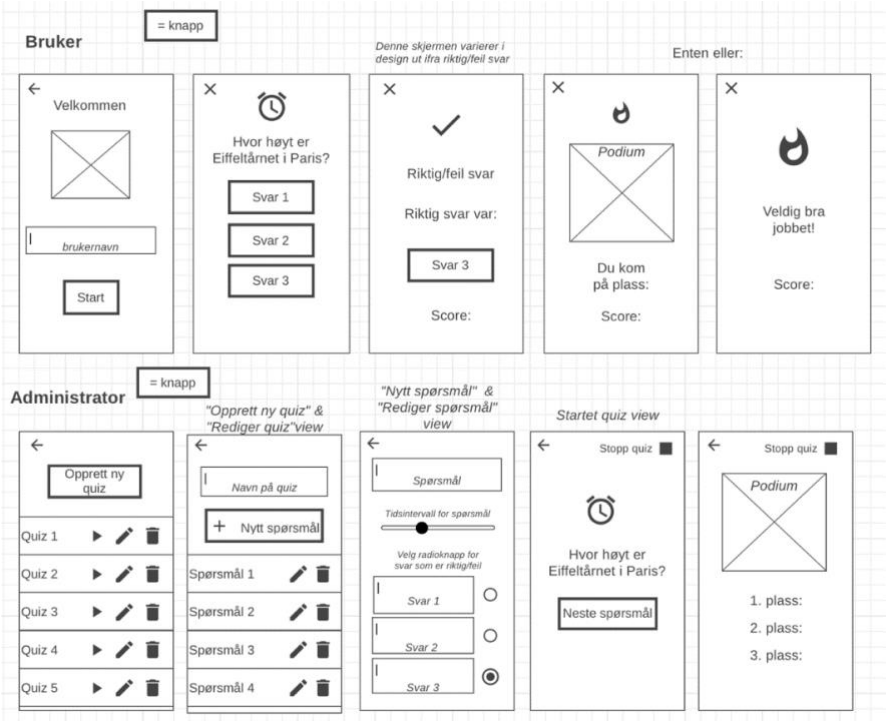
Domenemodellen, illustrert i Figur 12, viser overordnet de ulike domeneområdene som er koblet sammen via applikasjonen. Quizen er tiltenkt arrangementer som har som formål å sette søkelys på samfunnstematikk. Primært er det FNs bærekraftsmål som er gjeldende. Hver quiz har en Administrator som starter og styrer quizen. Dette vil være en ansatt i banken. Quizdeltagere tar del i en quiz ved bruk av nedlastet applikasjon på mobiltelefon.



Figur 12 – Domenemodell

Etter både funksjonelle krav, domeneoversikt og brukere var definert, designet teamet et førsteutkast av brukergrensesnitt for applikasjonen i form av wireframes, vist i Figur 13.

Disse ble laget basert på brukerreisene i kravdokumentasjonen. Det var viktig at wireframes illustrerte så mye av den ønskede funksjonaliteten som mulig slik at det var mulig for teamet å visualisere hvor mye som måtte utvikles. Deretter kunne funksjonalitet teamet tenkte var mulig å få utviklet trekkes ut fra wireframes inn i en prototype.



Figur 13 – Wireframes for brukergrensesnitt

Basert på wireframes ble det laget en mer detaljert skisse av hovedfunksjonaliteten i form an av en prototype i Figma. Prototypen ble utformet med tanken om at fokuset skulle ligge på å ferdigstille quizdeltager-siden av applikasjonen før administrator-siden. Det er derfor ikke laget prototype for administrator-siden. Administrator-siden viste seg etter hvert å være den delen teamet burde fokusere på først for å få en mer fullverdig MVP, og denne ble derfor utviklet i stedet for quizdeltager-siden vist i prototypen, illustrert i Figur 14.

Figur 14 – Prototype laget i Figma

4.2 Utviklingsprosessen

Som nevnt i kapittel 3.4.2, valgte teamet å ta utgangspunkt i Scrum som utviklingsmetodikk. Teamet har benyttet noen utvalgte artefakter fra metodikken. Målet med utviklingsprosessen var å produsere et MVP som demonstrerer hoveddelen av ønsket funksjonalitet.

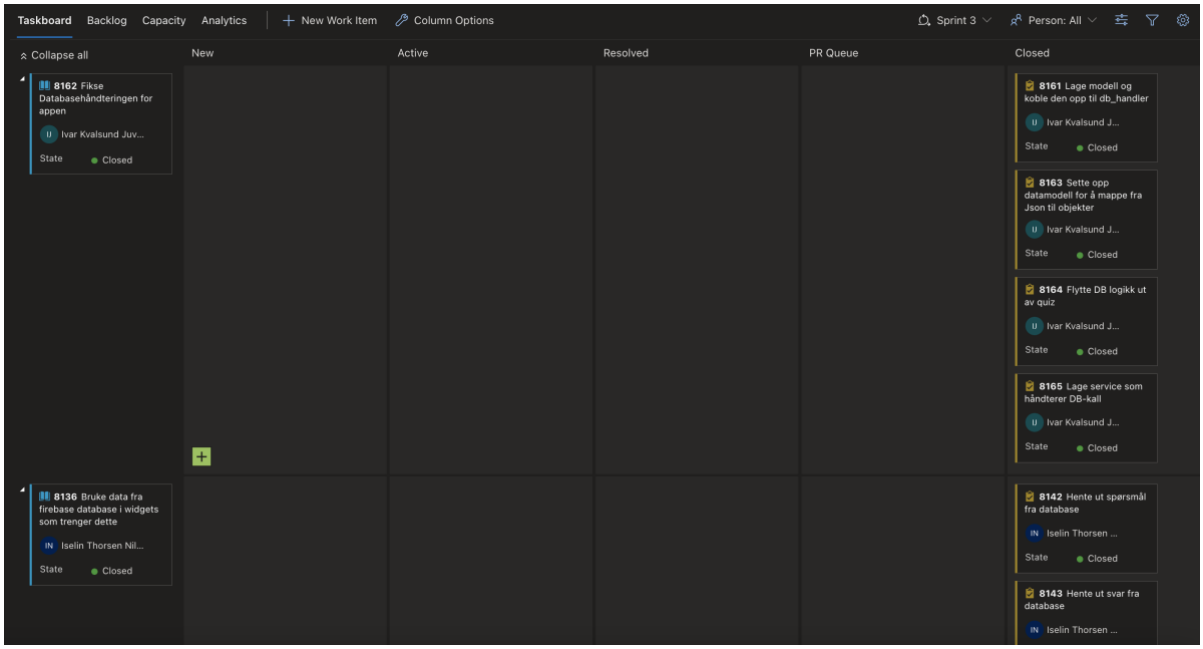
4.2.1 Sprint planlegging

Sprint planlegging er en systematisk planlegging av en kort periode med utvikling. Hver mandag i utviklingsperioden ble det gjennomført en sprint planlegging. Først definerte teamet den gjeldende sprintens mål; sprintmålet. Etter dette ble oppgavene for å nå sprintmålet fastsatt. På grunn av få medlemmer var det ikke fastsatt nøyaktig hvem som skulle gjøre hva, men en plukket heller oppgaver fra produkt og sprint backlog.

4.2.2 Sprint og kodeendringer

Hver sprint var en uke lang. Totalt ble fem sprinter gjennomført. Fast tidspunkt for jobbing med oppgaver i sprinten var tirsdag, onsdag og torsdag, hver uke, mellom klokken 08:00 og 15:00. Utenom dette jobbet hver enkelt selvstendig på egen tid. Underveis i sprinten jobbet en frem til backlog for den gjeldende sprinten var tom. Oversikten over oppgavene var oppført i en Kanban-tavle i Azure Devops. Der var det en oversikt over hvordan Teamet har hatt kontroll på hvem som jobber med hva, hvor mange oppgaver er fullført og

hvor mye som gjenstår. Det kommer også frem hvor langt på vei en har kommet med en oppgave. Denne Kanban-tavlen er vist i Figur 15.

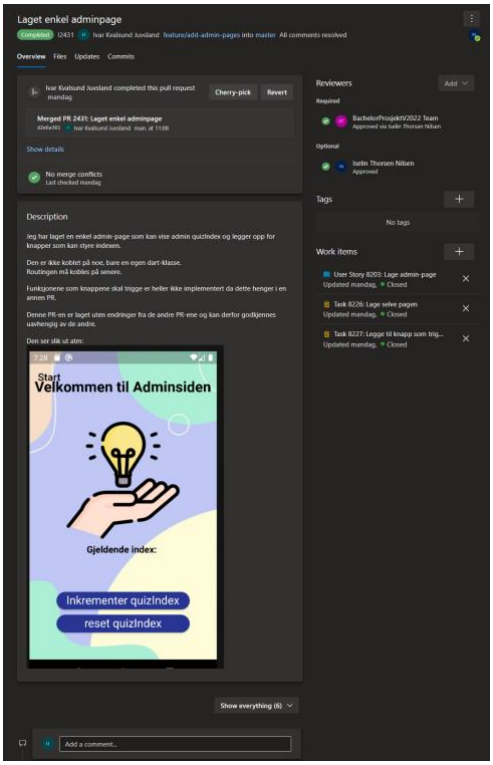


Figur 15 – KanBan-tavle i Azure DevOps

For å ha oversikt over endringer i koden har teamet benyttet verktøyene Git, Azure Devops og Fork for versjonskontroll. Ved bruk av disse har teamet kontinuerlig hatt PR for enhver endring i koden som er gjennomført i løpet av en sprint.

I en PR legges det inn forklaring, lenke til arbeidsoppgavene som er knyttet til kodeendringen og eventuelt bilder. Deretter må en annen i teamet lese gjennom kodeendringen og godkjenne den. Dersom den som godkjenner har spørsmål kan det legges inn kommentarer generelt eller direkte på en bestemt kodelinje. Det er da mulighet for diskusjon og forslag til endringer som dokumenteres i en PR. Figur 16 viser et eksempel på en PR som teamet gjennomførte i løpet av en sprint.

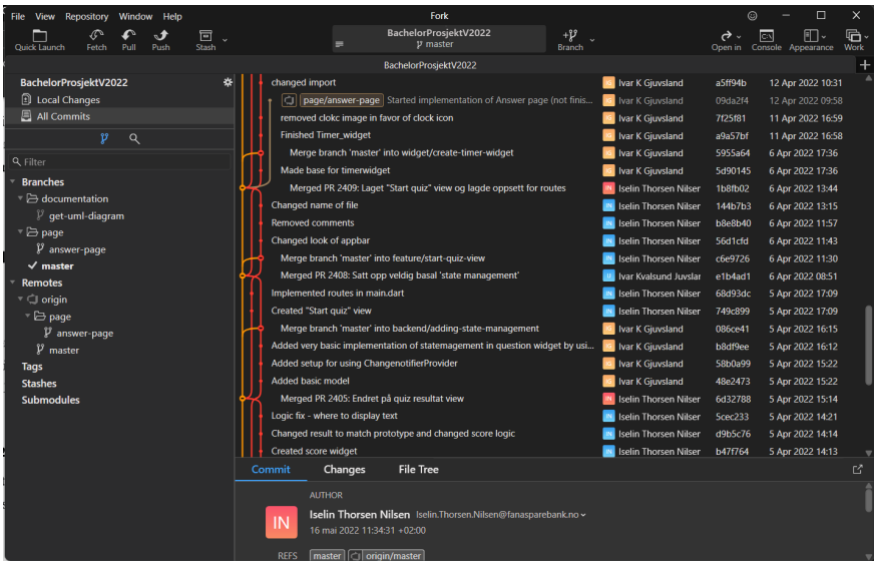
Ved avtalte endringer i kommentarer må medlemmet som lagde PR gjennomføre endringene, pushe opp den nye koden etterfulgt av at endringene må godkjennes av medlemmet som foreslo det. Når en PR blir godkjent blir kodeendringene flettet inn i en master-branch og oppgavene fra Kanban-tavlen knyttet til PR-en blir satt som «closed».



Figur 16 – PR i Azure DevOps

I tillegg til å hjelpe med å ha oversikt, medfører bruk av Git i Azure Devops en sikkerhet ved å ha flere kopier av en kodebase. Koden ligger lokalt hos alle parter i tillegg til i skyen hos Azure DevOps dersom noe skulle feile lokalt eller eksternt.

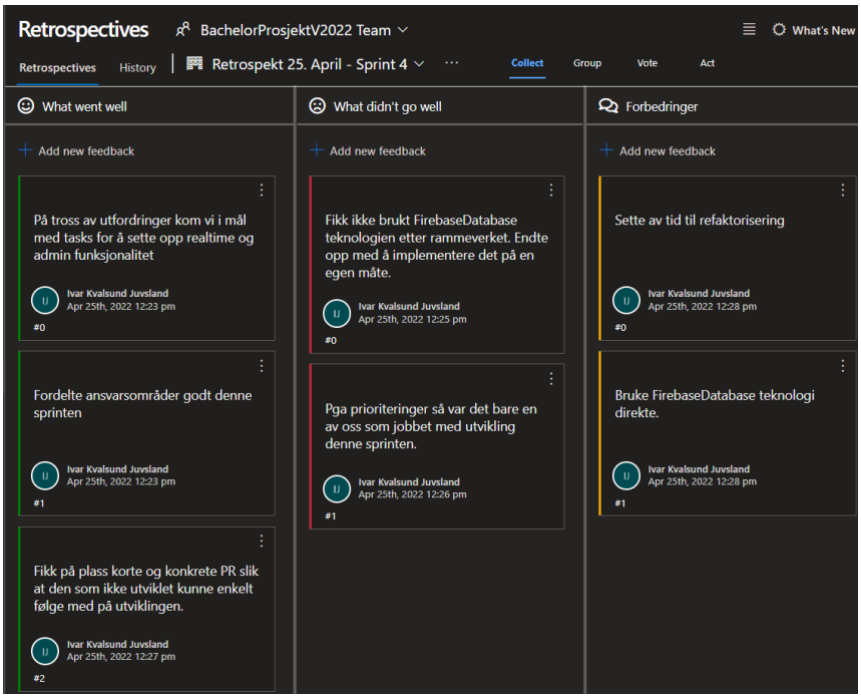
Git klienten, Fork, hjelper med å tydeliggjøre hvilken kodeversjon som kan ha forårsaket en feil, og hvilken versjon som en kan rulle tilbake til dersom en ny endring skaper uforutsette feil. I Figur 17 illustreres bruk av Fork og hvordan branch oversikten så ut i løpet av prosjektet.



Figur 17 – Fork Git-klient med branch oversikt

4.2.3 Retrospekt

Etter hver sprint ble det gjennomført et retrospekt. Hver mandag før sprintplanleggingen for neste sprint diskuterte teamet de ulike kolonnene i retrospekten. Her har det vært søkelys på «Hva gikk bra?», «Hva gikk dårlig?» og «Hva kan vi forbedre til neste sprint», vist i Figur 18. Dette har hjulpet teamet til å være mer bevisst på hvordan utviklingen gikk, samarbeid mellom medlemmene, og hva en kunne forbedre til neste sprint. På grunn av den korte utviklingsperioden tenkte teamet at det var viktig å lære tidlig av feil for å effektivisere perioden og bruke mest mulig tid på utviklingen av applikasjonen.



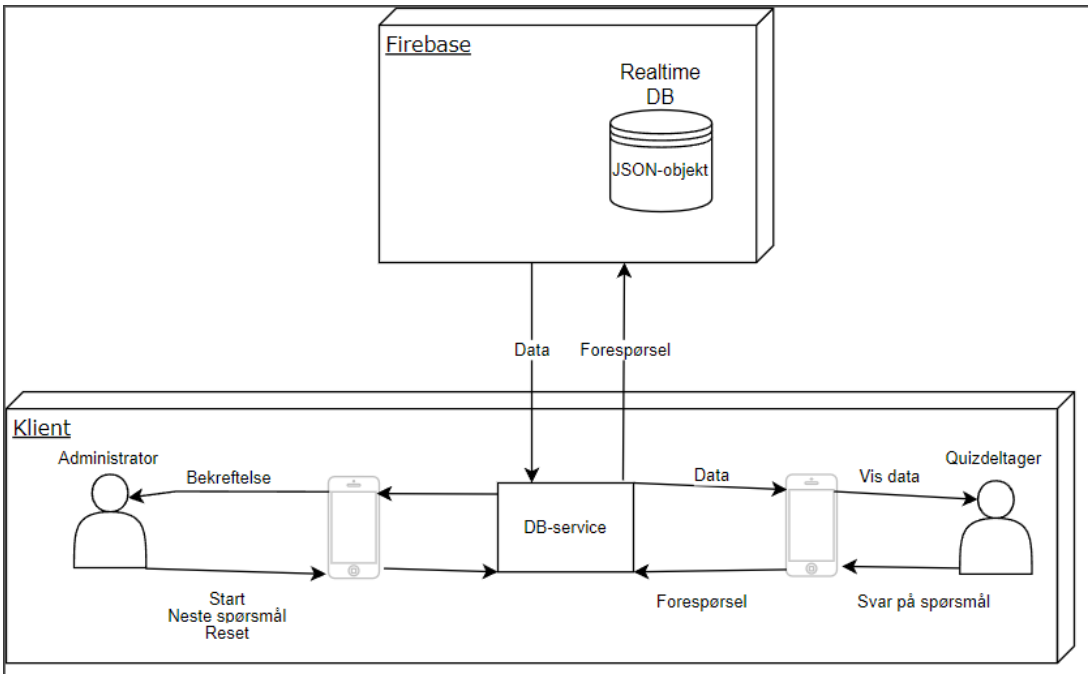
Figur 18 – Retrospekt tavle i Azure DevOps

4.3 Sluttprodukt

Resultatet av utviklingsprosessen er et fungerende MVP med de grunnleggende elementene fra kravspesifikasjonen. I dette kapittelet forklares programvarearkitekturen og brukeropplevelsen med tekniske forklaringer.

4.3.1 Programvarearkitektur

Programvarearkitekturen gir et overordnet bilde av hvordan programflyten i applikasjonen fungerer, illustrert i Figur 19. Hovedflyten kan beskrives ved at en klient spør om data fra Firebase databasen. Firebase sender data tilbake til klienten. Styringen for dette er gjennom DB-service som er knyttet til hver klient.



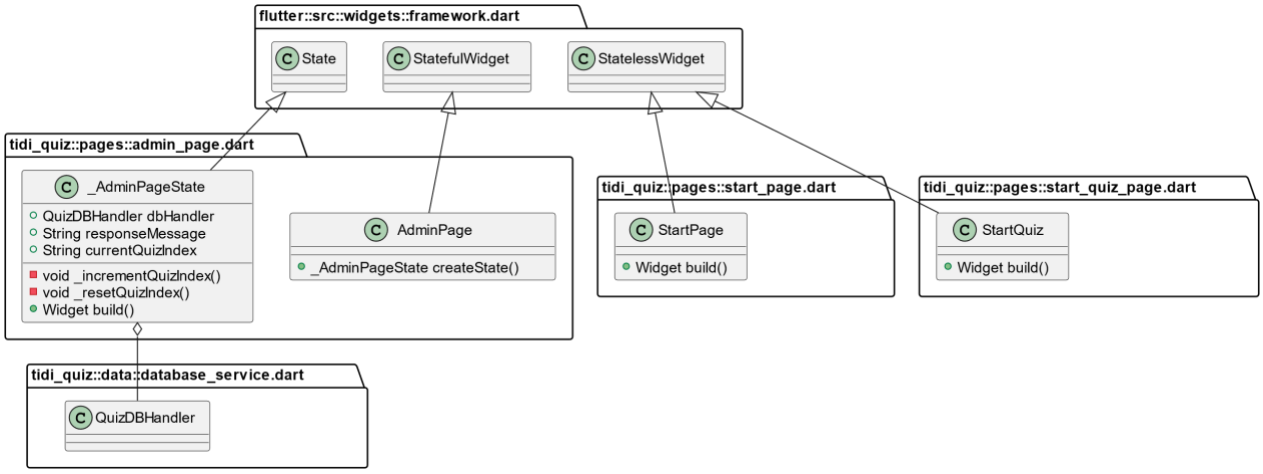
Figur 19 – Programvarearkitektur

4.3.2 Klient

Klienten inneholder nesten hele kodebasen. Den er delt opp i to forskjellige sider. Én for Administrator av quizen og én for quizdeltager.

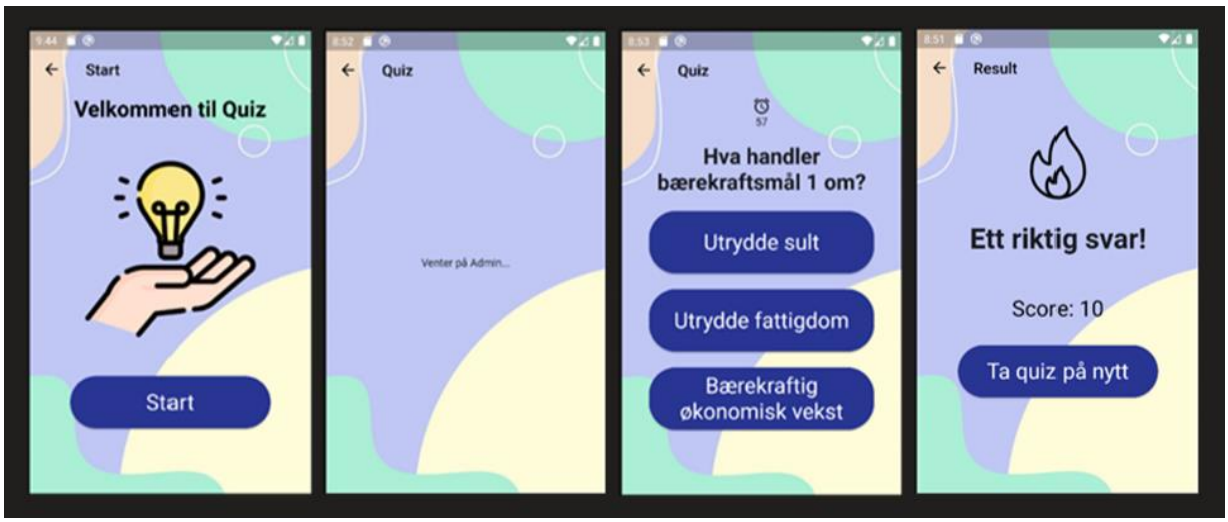
Når appen startes vil det komme opp en startside som ber brukeren om å velge hvilken brukertype de er, vist i Figur 20. Her kan det velges mellom «Quizdeltager» og «Admin». Ved interaksjon med hver av knappene blir brukeren dirigert til den korresponderende siden. Oversikten over de ulike sidene er tegnet opp i Figur 21.

Figur 20 – Startside i '2030'



Figur 21 – Klassediagram for de ulike sidene

Dersom en har valgt «Quizdeltager» blir en dirigert til «start_quiz_page». For å delta på quizen må en trykke på «Start»-knappen. Når deltageren har kommet til quiz-siden, vises en melding «Venter på Admin ...». Denne meldingen blir byttet ut med spørsmål og svaralternativer hentet fra databasen når administrator har startet quizen. Når en trykker på et svaralternativ, blir svaret kontrollert og score for resultat oppdatert i bakgrunnen. Deretter repeteres steget. Når deltageren har svart på alle spørsmål vil det komme opp en resultatvisning, som vist i Figur 22. Klassene som håndterer dette, er vist i klassediagram i vedlegg «Systemdokumentasjon».

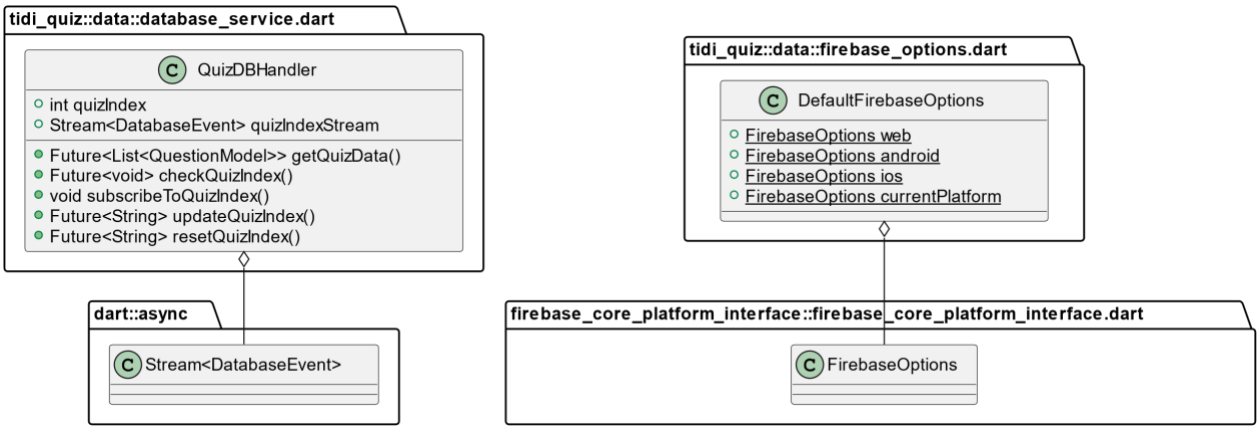


Figur 22 - Brukergrensesnitt for quizdeltager

Ved valg «Admin» blir en dirigert til «Admin_page». Der er det to alternativer for å styre quizen, som vist i Figur 23. «Reset quiz» setter spørsmål-indeks til «-1» og gjør at quizdeltagere ikke vil få opp spørsmål og svaralternativer. Det som skjer, er at klienten sender en oppdatering av databaseverdien for spørsmål-index. Dette vil klienter med quizdeltagere registrere og oppdatere visningen deretter.

«Neste spørsmål» vil enten starte quizen dersom den ikke er startet enda eller gå videre til neste spørsmål. Interaksjonen mellom klienter og database er den samme her som ved «Reset quiz». I klassediagrammet vist i Figur 24, kan en se metodene som er brukt av «database_service» for de ulike handlingene.

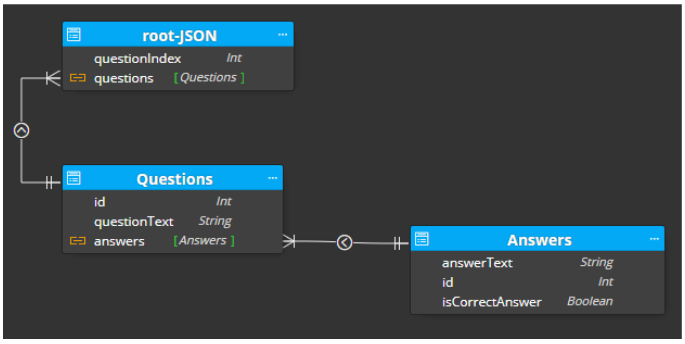
Figur 23 - Brukergrensesnitt for administrator



Figur 24 - Klassediagram for klasser med datahåndtering

4.3.3 Firebase

Databasen i Firebase er et produkt kalt «Realtime Database». Den inneholder ett enkeltstående JSON-tre vist i Figur 25. Her ligger all data som en quizdeltager eller administrator benytter seg av. Alle klientene er knyttet til den samme databasen, og derfor blir endringer av verdier oppfattet av alle som er tilkoblet. All interaksjon med databasen håndteres av «database_service» som er koblet opp mot Firebase «Realtime Database API». Dette skjer gjennom bruk av en programvareutvidelse for Flutter «Firebase_Core» og en tredjeparts kodepakke «firebase_database» (Google, 2022).



Figur 25 - Databasemodell

5 EVALUERING OG RESULTATER

Dette kapitelet tar for seg resultatene oppnådd ved hjelp av evalueringsmetodene beskrevet i underkapitlene.

5.1 Evalueringsmetode – Produkt

For å kunne svare på forskningsspørsmålet «*Hvordan påvirkes 4. klassingers resultat og engasjement i læring om bærekraft ved bruk av sanntidsquiz fremfor presentasjon?*» måtte teamet finne en passende evalueringsmetode.

For å kunne måle og sammenligne noen kategorier av kvalitative data (Grønmo, 2020) ønsket teamet å utføre en form for case studie på to grupper med 4. klassinger ved en lokal barneskole i Bergen (Wæhle, et al., 2020). De kvalitative kategoriene skulle inneholde målinger av hvor lærerik og hvor fornøyelig sesjonen hadde vært.

Teamet skulle ta en 4. klasse og dele denne inn i to mindre grupper på omtrent 10-15 personer. Én gruppe skulle gjennomføre en sanntidsquiz ved bruk av applikasjonen teamet

hadde utviklet, mens den andre skulle få høre på en presentasjon om temaet; bærekraft. Etter disse sesjonene, skulle gruppene med 4. klassinger svare på det samme spørsmålssettet for å kunne måle de kvalitative resultatene. Fra disse resultatene skulle teamet kunne foreta en teoretisk generalisering (Grønmo, 2020).

Basert på gruppen med 4. klassinger som skulle gjennomføre sanntidsquiz ved hjelp av applikasjonen, ønsket teamet også å utføre en simpel form av brukervennlighetstesting (Olsen, 2006). Denne gruppen skulle også gi en kort tilbakemelding til teamet om hvordan de syntes applikasjonen fungerte og om den var enkel nok å bruke.

Dette er overførbart til måten FSB ønsker å bruke applikasjonen på. Resultatet av dette kunne vært nyttig for å se hvilken effekt applikasjonen faktisk har på et segment av målgruppen deres. Resultatene kunne blitt tilpasset til å relatere til en presentasjon på et arrangement sammenlignet med bruk av sanntidsquiz applikasjonen.

5.2 Evalueringsresultat - Produkt

Evalueringen av produktet med 4. klassinger ble ikke mulig å gjennomføre innen tidsfristen. Dette ble for ambisiøst med tiden som stod til rådighet. Grunnet mye skiftende fokus i oppgaven ble ikke denne evalueringsmetoden prioritert tidlig nok. Årsakene til dette diskuteres i kapittel 6.3.

5.3 Evalueringsmetode – Vurderinger ved utviklingsmetodikk

På grunnlag av prosjektendringene måtte teamet velge en litt annen evalueringsmetode for å kunne svare på forskningsspørsmålet «*Hvilke vurderinger gjøres annerledes av to bachelorstudenter ved HVL i forhold til en etablert utviklergruppe fra FSB når det gjelder valg av utviklingsmetodikk for ett prosjekt?*».

Teamet ønsket å svare på forskningsspørsmålet ved hjelp av et semistrukturert muntlig intervju for å få frem ytterligere interessante svar fra intervjuobjektet (Andersen, 2020). Teamet skulle intervjue én ansatt fra FSB, originalt den tidligere kontaktpersonen, som var lederen til utviklergruppen.

En annen ansatt i FSB sin utviklergruppe endte opp med å være intervjuobjektet. Han er systemutvikler i FSB, og en del av utviklergruppen. Han har tatt over noen ansvarsområder

etter lederen for utviklergruppen sluttet, så det var derfor naturlig å spørre han om å stille som intervjuobjekt.

Teamet endte opp med å måtte endre litt på formatet av intervjuet med grunnlag i noen av prosjektets utfordringer, som diskuteres i kapittel 6.3. Intervjuet måtte bli gjennomført skriftlig for at intervjuobjektet fra FSB skulle få tid til å gjennomføre det, samt for at teamet skulle få tid til å analysere resultatene fra intervjuet. Intervjuet ble formatert i et digitalt skjema, Google Forms, med åpne spørsmål (Andersen, 2020), hvor intervjuobjektet skulle besvare så åpent og utfyllende som mulig, for å prøve å få tilnærmet like svar som i et semistrukturert intervju. Intervjuet i sin helhet er vedlagt i Vedlegg 2 - Intervjuresultat. Teamet forsøkte å lage spørsmålene så åpne som mulig, men med et formål om å kunne få svar på spesifikt hvilke vurderinger FSB har gjort ved valg av utviklingsmetodikk.

Med gjennomføringen av et skriftlig intervju var det mer oversiktlig for teamet å hente ut resultatene fra evalueringen. Det var også enklere for intervjuobjektet å ta den tiden som var nødvendig for å tenke gjennom og svare på spørsmålene. Teamet fikk da mer reflekterte og gjennomtenkte svar på disse.

Evalueringen av FSB sine vurderinger ved valg av utviklingsmetodikk gjorde at teamet kunne sammenligne resultatene med teamet sine egne vurderinger. På denne måten ønsket teamet å kunne besvare forskningsspørsmålet tilknyttet dette.

5.4 Evalueringsresultat – Vurderinger ved utviklingsmetodikk

Intervjuobjektet fikk tilsendt det digitale skjemaet med fem åpne spørsmål som var forventet noe lengre svar på. Han var ferdig med skjemaet etter rundt 40 minutter.

Oppsummert viser svarene fra intervjuresultatet at det er blitt gjort flere forskjellige vurderinger av FSB ved valg av utviklingsmetodikk for dette prosjektet, i kontekst av blant annet begrensinger og avgrensninger.

Først og fremst har FSB sin utviklergruppe en **etablert standard for utviklingsmetodikk** på prosjekter, som er DevOps. Det var derfor naturlig for dem at denne utviklingsmetodikken skulle bli brukt for dette prosjektet også.

Intervjuobjektet fra FSB skriver blant annet at «det viktigste med en utviklingsmetodikk er at den er fleksibel nok til å kunne håndtere flere utviklingssituasjoner pluss driften av en løsning. Altså at den ikke kommer i veien for å faktisk gjøre oppgavene». For FSB har det vært viktig å vurdere **drift aspektet** av alle deres eksisterende løsninger i tillegg til løsningen laget for dette prosjektet. Dette gjør «DevOps» metodikken meget relevant for dem, da «Ops»-delen av metodikken har like stort fokus som «Dev»-delen.

Utviklergruppen fra FSB er også splittet på de to ulike frontendene av applikasjonen; iOS og Android. Dette gjør at de har færre ressurser på de enkelte frontendene, som også påvirker utviklingstiden til produktet. Dette er en av vurderingene som FSB tar med ved valg av utviklingsmetodikk. Utfordringene rundt dette håndterer de med «den kontinuerlige kodeoppdateringen vi har - gjennom PR, kodegjennomganger og statusmøter» som er en del av utviklingsmetodikken – DevOps.

Det er også essensielt for FSB å vurdere teknologier og hvilken påvirkning dette har på prosjektene deres. For dette prosjektet, utviklingen og videreutviklingen av «Tidi», har det vært helt ny teknologi for alle utviklerne. Dette har påvirket tiden utviklingen har tatt fordi det ble lagt til ekstra tid før oppstarten av prosjektet til **opplæring**.

Det har også vært organisasjonsendringer som har påvirket utviklingen av deres versjon av produktet. Dette har påvirket ansvarsfordelingen og tiden produktet har tatt å utvikle. Det var da enda færre ressurser på utviklingen totalt, i tillegg til de forskjellige frontendene. Utviklergruppen har ikke endret utviklingsmetodikk grunnet dette, men det presiseres at **valget av metodikk burde ta høyde for nettopp slike utfordringer**, slik at stegene kan endres kontinuerlig.

Kodekontroll er også noe FSB har tatt med i vurderingene. Dette er viktig for å sikre minimering av feil og bugs etter systemet er rullet ut i produksjon. Det betyr at det skal være minst én annen utvikler som skal se over kodeendringen som er gjort, før den integreres inn i kildekoden til systemet. Dette må utviklingsmetodikken ta høyde og tilrettelegge for, og er en essensiell vurdering.

6 DISKUSJON

6.1 Sluttproduktet '2030'

Kravdokumentet definerer den ønskede funksjonaliteten. Sluttproduktet har de viktigste elementene på plass og er i seg selv et MVP. Produktet kan gjennomføre en sanntidsquiz med én administrator og flere quizdeltagere.

Det som mangler er primært opprettelse av ny quiz, nye spørsmål og mulighet for å endre den av en administrator som bruker klienten. Det er nå kun mulig å endre en quiz ved å logge inn i firebase og gjøre endringer i databasen. Videre mangler det innlogging og autentisering av administrator, brukernavn for quizdeltager, visning av riktig/galt svar, poengscore underveis og kåring av en vinner.

På tross av noen mangler av ønsket funksjonalitet, er det viktigste fundamentet for applikasjonen utviklet. Alle de funksjonskritiske kravene som å ha sanntid-tilkobling mot databasen, oppdatering av data fra administrator og quizflyt for quizdeltager er på plass. Derfor, med mer tid, ville teamet kunne utvikle de resterende funksjonelle kravene.

6.2 Vurderinger ved valg av utviklingsmetodikk

6.2.1 Forskjeller ved vurderinger

- **Etablert standard for utviklingsmetodikk** – Dette var noe FSB hadde som en av sine vurderinger, som var hovedgrunnen til at de ikke valgte ny utviklingsmetodikk for dette prosjektet. Teamet gjorde ikke denne vurderingen siden teamet ikke hadde en etablert utviklingsmetodikk fra før, som gjorde at valg av utviklingsmetodikk kunne skreddersys til dette prosjektet.
- **Driftsaspekt** – FSB må ta hensyn til at de må drifte eksisterende systemer og prosjekter som de har gående. Dette må utviklingsmetodikken tilrettelegge for. Denne vurderingen er ikke relevant for teamet sin del, siden teamet ikke har noen eksisterende systemer å drifte under utviklingen av dette prosjektet.
- **Ansvarsområder** – Mobilapplikasjonen som FSB skulle utvikle ble delt opp i to separate frontend, én for iOS og én for Android, som utviklerne var fordelt utover

på. Dette gjorde at prosessen måtte tilpasses, noe metodikken måtte tillate. Teamet har utviklet en felles kodebase for alle plattformer, med tilhørende backend, som begge medlemmene av teamet har tatt ansvar for. Dette var derfor heller ikke en vurdering teamet tok hensyn til.

- **Retrospekt** – Teamet var opptatt av at retrospekt skulle være en sentral del av utviklingsmetodikken. Det var en viktig vurdering for teamet at sprintene skulle evalueres og forbedres kontinuerlig gjennom utviklingsprosessen. Dette er ikke noe FSB sin utviklergruppe har vurdert eller gjennomført for dette prosjektet.

6.2.2 Likheter ved vurderinger

- **Opplæringstid** – Dette var noe både FSB og teamet hadde hatt som en vurdering. Det var ny teknologi for begge og det måtte derfor tas høyde for opplæringstid i oppstarten av prosjektet, noe som igjen utviklingsmetodikken måtte tillate.
- **Smidighet** – Det var en nødvendig vurdering for begge at utviklingsmetodikken var smidig, tillot at uforutsette hendelser kan skje og at justeringer kan bli gjort selv etter oppstarten av prosjektet.
- **Kodekontroll** – Denne vurderingen var også helt nødvendig for begge å ta med, siden kodekontroll er en helt essensiell faktor for at utviklingen av et produkt skal gå etter planen og bli gjennomført innen tidsfristen. Slik minimerer en risikoen for bugs i produktet som må fikses i ettertid av utrulling.

6.2.3 Diskusjon

Den opprinnelige planen for evalueringsmetode av FSB sine vurderinger, gjort ved valg av utviklingsmetodikk, var et semistrukturert muntlig intervju. Dette hadde totalt sett gitt et bedre evalueringsgrunnlag, fordi en får frem ytterligere interessante svar fra intervjuobjektet hvis en stiller åpne spørsmål. Det ble ikke mulig å gjennomføre dette som tiltenkt. Ved gjennomføring av et strukturert skriftlig intervju fikk teamet samlet inn nøkkelinformasjon om disse vurderingene som gjøres av FSB. Intervjuet ble heller ikke gjennomført med intervjuobjektet som originalt var tiltenkt, og teamet måtte finne et nytt mot prosjektets slutt. Konsekvensen av dette er at resultatet av evalueringen ikke er optimalt da evalueringsmetoden ikke kunne bli utført på best mulig måte.

Disse tilnærmingene påvirket resultatene i noen grad, men teamet mener at resultatene fra intervjuet likevel gav et godt evalueringsgrunnlag. Det gjør også at teamet klarer å svare på forskningsspørsmålet tilknyttet dette, i kapittel 7.1. På tross av dette kan ikke teamet trekke noen generaliserende konklusjoner fra resultatet av evalueringen. Det er ikke mulig å bruke resultatene utover å svare på det spesifikke forskningsspørsmålet tilknyttet prosjektet. Konklusjonene som er nådd gjelder kun for teamet sine to bachelorstudenter sammenlignet med en utviklergruppe fra FSB, for dette prosjektet. Dette er en begrensning med forskningsspørsmålet. Det kan i fremtiden være interessant å undersøke denne problemstillingen i større skala, for å få et større datagrunnlag å trekke konklusjoner fra.

6.3 Utfordringer

Først og fremst, har det vært flere større endringer av prosjektet etter prosjektets oppstart. Dette har medført noen utfordringer for teamet og skiftet oppgavens fokus flere ganger.

6.3.1 Tidsfrist og ansvar

Opprinnelig var det tiltenkt at teamet skulle utvikle «utvidelsen av Tidi-applikasjonen» helt på egen hånd og ha hele ansvaret for dette. Det var derfor naturlig at oppgaven skulle være en systemutviklingsoppgave, med mest fokus på produkt. Det ble gitt krav til bruk av teknologi for denne løsningen; .NET med SignalR, og Swift og Kotlin for frontendene. De funksjonelle kravene var også på plass. Med dette som utgangspunkt var teamet klar til prosjektets start i januar. Senere, etter de innledende møtene med både oppdragsgiver og veileder, informerte FSB om den korte tidsfristen produktet måtte være ferdigstilt innen. Det ble etter hvert klart for alle parter at dette produktet potensielt ble for stort for teamet å utvikle på egenhånd innen tidsfristen. Det var derfor da snakk om at kanskje noen andre utviklere fra FSB skulle bistå med utviklingen siden produktet måtte være ferdig innen gitt tidsfrist.

Veileder mente at tilleggsressurser fra FSB kunne bli komplisert, siden det måtte tydeliggjøres hva som var utført av teamet og hva som var utført av ressurser fra FSB. Idéen om en sammenligning av sluttprodukter oppstod da det ble mulig å utvikle to versjoner av det samme produktet. Teamet skulle da lage sin versjon av applikasjonen parallelt med FSB sin versjon av applikasjonen. Versjonene skulle utvikles isolert fra

hverandre, slik at det ikke skulle bli noen påvirkning. På denne måten fikk også teamet et helt separat produkt å utvikle på eget ansvar.

På dette tidspunktet var det begynt å bli dårlig tid til utviklingen av produktet på grunn av endringene og beslutningene som måtte tas. Teamet valgte på grunn av sammenligningen annen teknologi enn det som originalt var tiltenkt. Grunnet det valget ble det brukt tid på å undersøke alternativ passende teknologi som var forskjellig fra FSB. Valget falt da på teknologi som var helt ny og ukjent for teamet. Dette gjorde at en større periode av prosjektet ble satt av til opplæring. I tillegg tok utviklingen av produktet lengre tid enn antatt fordi det dukket opp nye problemer fortløpende som teamet aldri hadde støtt på før.

6.3.2 Sammenligning av sluttprodukter

For å kunne sammenligne sluttprodukter var planen å utføre brukertesting på den samme brukergruppen som FSB hadde tiltenkt. På denne måten kunne teamet gjennomført en evaluering av applikasjonen og samtidig fått tilbakemeldinger fra de samme brukerne om forskjellene og likhetene mellom sluttproduktene. Disse tilbakemeldingene kunne teamet fått ved hjelp av en enkel spørreundersøkelse (Andersen, 2020). Teamet ønsket å samle resultatene fra spørreundersøkelsen for å kunne undersøke forskjellene og likhetene mellom sluttproduktene.

Både teamet og FSB fikk de samme funksjonelle kravene fra start og ville derfor vurdere hvilken versjon av applikasjonen som var mest brukervennlig, innfridde mest funksjonelle krav og som fungerte til sitt formål. Målet var til slutt å sammenligne resultatene av dette.

Mot slutten av prosjektet ble teamet informert om at FSB ikke kom til å bli ferdige med sitt sluttprodukt innen tidsfristen. FSB sin versjon ble så vidt begynt på rett før teamet sin utviklingsperiode var over, og når teamet var ferdig med sitt sluttprodukt hadde ikke FSB noe funksjonalitet på plass. Det var derfor ikke noe alternativt produkt for teamet å sammenligne produktet sitt med. Hovedfokuset til prosjektet måtte derfor skiftes over til produktet som var utviklet av teamet og utforskingen av utviklingsmetodikker.

Teamet ønsket å få på plass en form for brukertesting likevel, men med den aktuelle målgruppen, elever fra en 4. klasse. Dette var mot slutten av prosjektet og det ble for liten tid til å koordinere og gjennomføre dette, samt analysere resultatene fra det.

En av grunnene til at det ble forsinkelser hos FSB er at prosjektleder, som var leder for FSB sin utviklergruppe, sluttet i FSB midt i prosjektløpet. Dette påvirket flere deler av prosjektet for både teamet og FSB sin del.

6.3.3 Sluttprodukt

Det er forskjellige grunner til at applikasjonen mangler funksjonalitet. Blant annet var ny teknologi en større utfordring enn forventet. Dette gjorde at det måtte settes av mer tid til opplæring, og utviklingen av applikasjonen ble derfor ytterligere forsinket. Dersom teamet ikke hadde valgt ukjent teknologi for å kunne gjennomføre sammenligningen av sluttprodukter, hadde teamet utviklet med kjent teknologi og hadde ikke trengt like lang tid til opplæring.

I tillegg er Dart og Flutter relativt ung teknologi, og har hatt større endringer siden de ble først gitt ut. Dette medførte at en del av opplæringen som ble gjennomført før teamet startet på utviklingen fokuserte på måter å utvikle funksjonalitet på som det ikke lenger var støtte for. Noen implementasjoner av funksjonalitet fungerte også på en annen måte, hvor dette ikke alltid var like godt dokumentert. Dette førte til at noen oppgaver fra en sprint ble flyttet over til neste, som skapte en forskyving av oppgaver og dermed forsinkelser.

6.4 Refleksjoner

Den avgjørende faktoren for teamet har vært tid. Kombinasjonen av dette og helt ny teknologi gjorde at teamet brukte mer tid enn forventet på utviklingstiden og kom ikke like langt med utviklingen av produktet som planlagt. Hvis teamet hadde hatt mer tid til utvikling av produktet hadde nok teamet fått på plass mer funksjonalitet utover MVP.

Risikoanalyse og planlegging av tiltak som må iverksettes burde blitt høyere prioritert. Risikomomenter som teamet identifiserte, men ikke var særlig bekymret for, skjedde og det ble travelt for teamet å komme opp med alternative løsninger som ikke var tenkt på tidligere. Teamet ser nå viktigheten av å legge inn mer tid og energi i å gjennomføre en mer uttømmende risikoanalyse, med bedre forhåndsdefinerte tiltak.

Ansvarsfordeling av utviklingsområder var ikke noe teamet så på som et alternativ siden teamet kun består av to utviklere. Dette er likevel noe som blir benyttet en del i

arbeidslivet, hvor et utviklerteam deles opp til å jobbe med enten frontend eller backend, eller med begge. Det har vært nyttig for teamet å erfare at dette kanskje er noe som kunne vært lurt å få til i et fremtidig prosjekt, da en kan erfare at dette kanskje er mer effektivt i gjennomføringen av prosjekter.

Teamet innser også de utfordringene som har oppstått i prosjektet er noe som skjer også i arbeidslivet. Utsettelser, større endringer og uforventet arbeidsmengde er realiteten til de aller fleste utviklere, noe som FSB demonstrerte i dette prosjektet. Det som er viktig å ta med seg videre fra dette er at en må tilrettelegge for at disse tingene kommer til å skje og ha gode strategier for å håndtere det. Dette er noe medlemmene av teamet kommer til å ta med seg videre.

7 KONKLUSJON OG VIDERE ARBEID

7.1 Konklusjon

Problemstillingene beskrevet i kapittel 1.5 er utforsket gjennom prosjektperioden. Metodene teamet endte opp med å bruke for å utforske disse har gitt varierende resultater, og kun gitt mulighet til å svare fullstendig på én av dem.

1. Hvordan kan teamet utvikle en sanntidsquiz-applikasjon som bidrar til læring om bærekraft blant barn og unge?

Applikasjonen er utviklet som forklart i kapittel 4, men ikke brukertestet. Det gjør at teamet ikke kan besvare forskningsspørsmålet «*Hvordan påvirkes 4. klassingers resultat og engasjement i læring om bærekraft ved bruk av sanntidsquiz fremfor presentasjon?*». Likevel har teamet utforsket problemstillingen og funnet ut av hvordan et team kan utvikle en sanntidsquiz-applikasjon ved hjelp av Flutter og Dart med Firebase, som ble beskrevet i kapittel 4.

2. Hvordan kan teamet utforske vurderinger som er gjort av to bachelorstudenter i forhold til en etablert utviklergruppe ved valg av utviklingsmetodikk?

Den andre problemstillingen ble utforsket i resultatene fra evalueringen av vurderinger ved valg av utviklingsmetodikk, i kapittel 5.4.

Det tilhørende forskningsspørsmålet «*Hvilke vurderinger gjøres annerledes av to bachelorstudenter ved HVL i forhold til en etablert utviklergruppe fra FSB når det gjelder valg av utviklingsmetodikk for ett prosjekt?*» er diskutert i kapittel 6.2. Vurderingene som var annerledes er:

- Etablert standard for utviklingsmetodikk
- Drift av eksisterende løsninger
- Fordelte ansvarsområder i utviklingen av den samme mobilapplikasjonen
- Ikke bruk av retrospekt

Vurderingene som er annerledes går hovedsakelig på hensyn som en bachelorgruppe i et enkeltstående prosjekt ikke trenger å tenke på.

Resultatene viser at teamet kan konkludere med at det finnes både forskjeller og likheter i vurderinger som er gjort ved valg av utviklingsmetodikk mellom to bachelorstudenter ved HVL og én utviklergruppe fra FSB.

7.2 Videre arbeid

Store deler av applikasjonen er ferdig så videre arbeid ville basert seg på å fullføre de funksjonelle kravene fra kravspesifikasjonen. Dersom alle kravene hadde kommet på plass, vil en i praksis ha en fullt fungerende sanntidsquiz-applikasjon med mulighet for redigering av quiz direkte i appen. Da kunne en publisert applikasjonen i TestFlight, som hadde muliggjort brukertesting av produktet med flere grupper. En kunne da undersøkt effekten applikasjonen har på læring og engasjement hos barn og unge. Etter brukertesting og eventuelle forbedringer kunne en publisert applikasjonen i Google Play og App Store.

Kodebasen kan også forbedres. Siden teamet har liten erfaring med både Flutter og Dart er det nok en del forbedringspotensial som kan fikses ved å bli mer erfaren med hvordan Flutter og Dart fungerer.

Videre kunne en skalert løsningen slik at applikasjonen kan brukes av flere grupper og administratorer samtidig. Da måtte en sett på bruken og implementasjonen av flere individuelle sesjoner i sanntidsdatabasen i Firebase.

Enhetstester er også noe som burde bli laget for de ulike funksjonene i applikasjonen. Dette er essensielt å få på plass dersom løsningen skal utvides og det er flere komponenter som skal snakke sammen på komplekse måter. Hvis applikasjonen skal publiseres er det også relevant å se på generell testing ved hjelp av kode før en går over på brukertesting.

7.2.1 Relevans av resultater for andre

For **Høgskulen på Vestlandet** kan det være interessant å se hvilke vurderinger deres bachelorstudenter ved informasjonsteknologi- og dataingeniørstudiene gjør ved valg av utviklingsmetodikk. Dersom dette blir undersøkt nærmere i et større forskningsprosjekt kan det være relevant for utformingen av studiene videre.

Prosjektresultatene kan si noe om **hvilket grunnlag studentene har** for å gjøre disse vurderingene, som er viktige vurderinger å kunne gjøre i arbeidslivet. Forskriften om rammeplan for ingeniørutdanning sier at «den skal sikre at utdanningene forholder seg til de standarder og kriterier som gjelder for ingeniørutdanning, og imøtekommer samfunnets nåværende og framtidige krav til ingeniører» (NORSK LOVTIDEND, Avd. I Lover og sentrale forskrifter mv., 2018). Vurderinger studentene ved informasjonsteknologi- og dataingeniørstudiene er i stand til å gjøre ved valg av utviklingsmetodikk er bare et snevert aspekt av dette. Likevel, er den kompetansen som ligger til grunn for å gjøre disse vurderingene noe som bygges opp gjennom hele studieløpet ved å forstå helheten av sitt fagfelt.

Prosjektresultatene kan også være relevante for **FSB** å se på, da det kan gi dem **ytterligere innsikt i vurderinger** som kan gjøres. Det kan fungere som en form for refleksjon for dem, og gi dem mulighet til å se på eventuelt andre utviklingsmetodikker som kunne passet dem bedre. Noe som hadde vært spesielt relevant for dem å se på er **bruken av retrospekt** og om dette er noe som de kan implementere i deres bruk av utviklingsmetodikk.

Applikasjonen teamet har utviklet er også noe FSB sin utviklergruppe kunne tatt inspirasjon fra. Kanskje det hadde vært lurere for dem å bruke **multi-plattform teknologi** for å få utviklingsprosessen til å gå litt hyppigere siden alle utviklerne kunne jobbet mot den samme kodebasen, samtidig som utviklingen i seg selv kunne blitt smidigere.

8 REFERANSER

Biørn-Hansen, A., 2020. *An empirical investigation of performance overhead in cross-platform mobile development frameworks,* s.l.: Springer.

Wæhle, E., Dahlum, S. & Grønmo, S., 2020. *Store Norske Leksikon.* [Internett]
Available at: https://snl.no/case-studie
[Funnet 23 April 2022].

Grønmo, S., 2020. *Store Norske Leksikon.* [Internett]
Available at: https://snl.no/kvalitativ_metode
[Funnet 23 April 2022].

Olsen, Ø., 2006. *Praktisk brukertesting,* Oslo, Kongsvinger: Statistisk sentralbyrå.

Orgeret, K. S., 2018. *Store Norske Leksikon.* [Internett]
Available at: https://snl.no/intervju
[Funnet 23 April 2022].

Andersen, G., 2020. *NDLA - Kvalitative intervjuundersøkelser.* [Internett]
Available at: https://ndla.no/nb/subject:1:9bb7b427-3f5b-4c45-9719-efc509f3d9cc/topic:1:432baee9-5671-47ce-870e-48b8fc3b7a42/topic:1:1db7bf3c-3a7b-44af-b632-e3c5ff2a999e/resource:201ce19e-7011-49a6-b415-91fd42d5dfe9
[Funnet 23 April 2022].

Andersen, G., 2020. *NDLA - Spørreskjema.* [Internett]
Available at: https://ndla.no/nb/subject:1:9bb7b427-3f5b-4c45-9719-efc509f3d9cc/topic:1:432baee9-5671-47ce-870e-48b8fc3b7a42/topic:1:1db7bf3c-3a7b-44af-b632-e3c5ff2a999e/resource:e2c1dd82-020e-4845-8215-7ae1ff3b422f
[Funnet 23 April 2022].

GEAMBAŞU, C. V., JIANU, I., JIANU, I. & GAVRILĂ, A., 2011. *INFLUENCE FACTORS FOR THE CHOICE OF A SOFTWARE DEVELOPMENT METHODOLOGY,* Bucharest: The Bucharest Academy of Economic Studies, Romania.

Mishra, A. & Dubey, D., 2013. A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios. *International Journal of Advance Research in Computer Science and Management Studies,* 1(5), p. 7.

Wang, A. I. & Tahir, R., 2020. The effect of using Kahoot! for learning. *The effect og using Kahoot! for learning - A litterature review,* 6 Februar.

Nugroho, S., Waluyo, S. H. & Hakim, L., 2017. Comparative Analysis of Software Development Methods between Parallel, V-Shaped and Iterative. *International Journal of Computer Applications ,* 169(11), p. 5.

Dora, S. K. & Dubey, P., 2013. SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC) ANALYTICAL COMPARISON AND SURVEY ON TRADITIONAL AND AGILE METHODOLOGY. *NATIONAL MONTHLY REFEREED JOURNAL OF RESEARCH IN SCIENCE & TECHNOLOGY,* 2(8), p. 9.

Westerlund, A. et al., 2015. Digital casts in orthodontics: A comparison of 4 software systems. *American Journal of Orthodontics and Dentofacial Orthopedics,* 147(4), p. 7.

Despa, M. L., 2014. Comparative study on software development methodologies. *Database Systems Journal,* V(3), p. 21.

Abrahamsson, P., Salo, O., Ronkainen, J. & Warsta, J., 2002. *Agile Software Development Methods: Review and Analysis,* Espoo, Finland: Abrahamsson, P., Salo, O., Ronkainen, J. & Warsta, J..

Balaji, S. & Murugaiyan, D., 2012. WATEERFALLVs V-MODEL Vs AGILE: A COMPARATIVE STUDY ON SDLC. *International Journal of Information Technology and Business Management,* 2(1), p. 5.

Coplien, J. & Bjørnvig, G., 2010. *Lean Architecture for Agile Software Development.* 1 red. Chichester, England: Wiley.

Ebert, C., Abrahamsson, P. & Oza, N., 2012. Lean Software Development. *IEEE Software,* 29(5), pp. 22-25.

NORSK LOVTIDEND, Avd. I Lover og sentrale forskrifter mv., 2018. *Lovdata.* [Internett]

Available at: https://lovdata.no/dokument/LTI/forskrift/2018-05-18-870
[Funnet 15 Mai 2022].

Dart Dev, 2021. *Dart.* [Internett]
Available at: https://dart.dev/articles/libraries/creating-streams
[Funnet 15 Mai 2022].

Ebert, C., Gallardo, G., Hernantes, J. & Serrano, N., 2016. DevOps. *IEEE Software*, Juni, pp. 94-100.

Kahoot! AS, 2022. *What is Kahoot!?.* [Internett]
Available at: https://kahoot.com/what-is-kahoot/
[Funnet 11 Mars 2022].

Microsoft, 2022. *ASP.NET Core SignalR supported platforms.* [Internett]
Available at: https://docs.microsoft.com/en-us/aspnet/core/signalr/supported-platforms?view=aspnetcore-6.0
[Funnet 16 Mai 2022].

GitLab, 2021. *Statista.* [Internett]
Available at: https://www.statista.com/statistics/1233917/software-development-methodologies-practiced/
[Funnet 21 Mai 2022].

FN-SAMBANDET, 2022. *FNs bærekraftsmål.* [Internett]
Available at: https://www.fn.no/om-fn/fns-baerekraftsmaal
[Funnet 20 Mai 2022].

Martín-Sómer, M., 2021. *ScienceDirect.* [Internett]
Available at: https://www.sciencedirect.com/science/article/pii/S174977282100035X
[Funnet 30 Mars 2022].

Pranczke, B., 2021. *Netguru.* [Internett]
Available at: https://www.netguru.com/blog/lean-software-development
[Funnet 27 Februar 2022].

Jeffries, R., 2018. *The Nature of Software Development.* [Internett]
Available at:

https://kupdf.net/download/pragmaticthenatureofsoftwaredevelopment1941222374pdf_5a5bb59ae2b6f58a383f2edd_pdf
[Funnet 22 Februar 2022].

Google, 2022. *Firebase.* [Internett]
Available at: https://firebase.google.com/docs/database
[Funnet 3 Mars 2022].

Mishra, L., 2021. *LinkedIn.* [Internett]
Available at: https://www.linkedin.com/pulse/flutter-firebase-killer-combination-mobile-app-lalit-mishra
[Funnet 15 Mars 2022].

Google, u.d. *Flutter.* [Internett]
Available at: https://flutter.dev/multi-platform
[Funnet 3 Mars 2022].

Schwaber, K., 2020. *Scrum Guides.* [Internett]
Available at: https://scrumguides.org/scrum-guide.html
[Funnet 22 Februar 2022].

Microsoft, 2022. *Microsoft Docs.* [Internett]
Available at: https://docs.microsoft.com/en-us/aspnet/core/signalr/introduction?WT.mc_id=dotnet-35129-website&view=aspnetcore-6.0
[Funnet 9 Februar 2022].

Google, 2022. *Firebase core for Flutter.* [Internett]
Available at: https://pub.dev/packages/firebase_core
[Funnet 14 Mai 2022].

Google, 2022. *Firebase database plugin.* [Internett]
Available at: https://pub.dev/documentation/firebase_database/latest/
[Funnet 26 April 2022].

Andersen, G., 2020. *NDLA.* [Internett]
Available at: https://ndla.no/subject:1:54b1727c-2d91-4512-901c-

8434e13339b4/topic:2:432baee9-5671-47ce-870e-48b8fc3b7a42/topic:2:1db7bf3c-3a7b-44af-b632-e3c5ff2a999e/resource:e2c1dd82-020e-4845-8215-7ae1ff3b422f [Funnet 4 Mai 2022].

9 VEDLEGG

9.1 Vedlegg 1 – Risikoanalyse

	Hendelse/Risiko	Årsak	Sannsynlighet	Konsekvens	Risikoprodukt	Tiltak
1	*Applikasjonen blir ikke tatt i bruk*	*Ineffektiv og utilstrekkelig funksjonalitet*	Middels(3)	Høy(4)	12	*Overholde tidsfrister og allokere nok ressurser på prosjektet. Bruke god tid på planlegging og kartlegging i forkant.*
2	*Applikasjonen blir ikke ferdig*	*Ikke nok ressurser og/eller ikke gode nok ressurser*	Høy(4)	Høy(4)	16	*Overholde tidsfrister og allokere nok ressurser på prosjektet. Bruke god tid på planlegging og kartlegging i forkant.*
3	*Misbruk av personinformasjon*	*Ikke god nok sikkerhet rundt personvern*	Middels(3)	Svært høy(5)	15	*Sikre at applikasjonen følger compliance for personvern og annen sikkerhet.*
4	*Feil/bugs i applikasjonen etter utrulling*	*Ikke god nok testing/bruker testing*	Høy(4)	Høy(4)	16	*God nok tid og nok ressurser på testing og brukertesting.*
5	*Sammenligning av produkter vanskelig å gjennomføre*	*Ikke konkret nok hvordan sammenligningen skal gjennomføres*	Middels(4)	Svært høy(5)	15	*Konkretisere og tenke godt gjennom sammenligningsgrunnlaget på forhånd. Sette av nok tid til sammenligning og analyse av hvert enkelt produkt.*
6	*Produktene som skal sammenlignes er for like*	*Produktene er så like at sammenligningen gir lite resultater*	Svært høy(5)	Svært høy(5)	25	*Utviklingen av begge produktene skal være isolert for det andre teamet for å forsøke å unngå dette. Naturligvis vil oppgavens art gi like produkt, men på grunn av forskjellig teknologi kan de være ulike på andre punkter.*
7	*For lite å sammenligne*	*Det er ikke nok å skrive om/undersøke for å få en god sammenligning*	Middels(3)	Svært høy(5)	15	*På forhånd definere nok sammenligningsmateriell og heller spisse det mer inn senere hvis teamet ser at det blir for mye å skrive om. Se bredt på alle aspekter ved utvikling og produkt.*
8	*Tekniske problemer*	*Tekniske problemer som utsetter/stopper utvikling*	Middels(3)	Høy(4)	12	*Starte i god tid med oppsett og utvikling for å ta høyde for disse problemene. Spørre raskt om hjelp ved behov.*
9	*Sykdom*	*Noen blir syke over lengre tid/utilgjengelig*	Middels(3)	Lav(2)	2	*Fordele oppgaver godt, samtidig som hver enkelt på teamet har god kontroll på hva den andre parten holder på med grunnet teamets størrelse. Hvis en blir utilgjengelig tar den andre over arbeidsoppgaver i den perioden.*

9.2 Vedlegg 2 - Intervjuresultat

Bachelorprosjekt - Utforsking av utviklingsmetodikk

Dette skjemaet er laget for å spørre utviklerne i FSB om hvordan de har gjennomført sin utviklingsmetodikk. NB! Alle spørsmålene er spesifikke for "Tidi" app prosjektet.

Spørsmål: "Har dere en spesifikk utviklingsmetodikk dere følger? Hvis ja, forklar hvordan dere benytter den"

Utviklingsmetodikk = arbeidsmetode for utvikling. (feks. Scrum, DevOps)

Svar: «Ja, vi bruker en versjon av DevOps metoden. Jeg skriver versjon, fordi alle utviklingsmetodikker blir litt tilpasset til gruppen som bruker den. Men i bunn ligger DevOps konseptet.

Før hver større kodeendring vi setter i gang med, er det en planleggingsfase. Her kartlegger vi hva som ønskes endres, og tar eventuelle avklaringer med bestiller (hvis det finnes en). Dette er aldri likt. Noen ganger er det for eksempel et ønske fra en annen avdelingen i banken, andre ganger er det en bug som må fikses. Planleggingen avhenger derfor veldig i forhold til dette. Men, vi etterstreber alltid å dokumentere avklaringer og avgjørelser vi har tatt i en kravspesifikasjon. I tillegg legger vi opp kort i en prosjekttavle. Disse kortene beskriver oppgavene som skal gjøres og er rangert etter størrelse (epics, fetures, user stories, tasks).

Når planleggingen er gjort begynner vi på koden. Da har vi ofte satt en prioritering av hvilke oppgaver som skal utføres først. På større kodeendringer blir oppgavene fordelt ut over flere utviklere, slik at vi kan jobbe parallelt med de. For hver Task – og av og til for hver User Story – lages det en pull request. Disse må godkjennes av minst to andre utviklere.

Vi har også CI/CD bygg som kjøres ved hver gjennomførte Pull Request. Dette lar oss følge med på enhetstester og slikt. Vi tester også løsningene manuelt jevnlig i utviklingsprosessen.

Det er ikke alltid vi ruller ut koden for hver gjennomførte PR / for hvert suksessfulle CI-bygg. Dette er noe vi gjør stort sett på løsninger som allerede er satt i produksjon. På nye løsninger venter vi stort sett til det som tilsvarer en major release før vi ruller ut koden. Dette varierer nok kanskje fra andre DevOps team.

Når koden er rullet ut begynner såklart Ops delen av tjenesten. Vi lager tjenestene våre med varslings muligheter hvis noe skulle gå gale. Skulle en feil skje går det eposter til alle utviklerne, som beskriver feilen som har skjedd og i hvilken løsning det skjedde i.

Når det kommer ønsker om endringer på eksisterende løsninger, kjører vi hele denne prosessen på nytt. Det skrives en kravspec, det settes opp kort, det utvikles, det testes, og det rulles ut."

Spørsmål: "Hva tenker du er det viktigste i en utviklingsmetodikk?"

Svar: "Det viktigste med en utviklingsmetodikk er at den er fleksibel nok til å kunne håndtere flere utviklingssituasjoner pluss driften av en løsning. Altså at den ikke kommer i veien for å faktisk gjøre oppgavene. Det er ugunstig med en metodikk som for eksempel tar mye tid vekk fra å løse problemet eller skrive koden. Metodikken må da være smidig og lett å forstå."

Spørsmål: "Hvilke vurderinger har dere gjort ved valg av utviklingsmetodikk?"

Feks: «Vi er et utviklingsteam som ikke bare har oppgaver knytt til dette prosjektet, vi har også driftsoppgaver. Derfor er det naturlig å ta dette med i vurderingen av utviklingsmetodikk.»

Svar: "Å kunne ta høyde for både utvikling- og driftsoppgaver stod veldig sentralt i valget av metodikk. Dette føler jeg at DevOps er godt egnet til. Når det er sagt ble det ikke gjort et eget valg av metode før utviklingen av Tidi-appen, verken fase en eller to. Alle utviklerne hadde allerede god kjennskap til DevOps, så det falt naturlig å fortsette med denne for dette prosjektet."

Spørsmål: "Hvilke begrensinger har vært med i vurderingen ved valg av utviklingsmetodikk?"

Dette gjelder kun for dette prosjektet. Begrensninger = ressurser som dere ikke kan påvirke (feks. teamstørrelse, tid, kunnskap)

Svar: "Tidi-app prosjektet hadde noen unike utfordringer som vi ikke ellers har vært borti. Dette gjelder for så vidt begge fasene av app-utviklingen. Den første fasen er da appen ble laget. Fase to er den nåværende fasen, og gjelder quiz-videreutviklingen av appen.

Den største utfordringen var det var ny teknologi for nesten alle utviklerne. I fase en var det ingen av utviklerne som hadde laget en app for FSB før, verken på IOS eller Android. Det betydde at en del av prosjekttiden gikk på å lære seg dette. Det ble derfor en utvidet startfase av prosjektet, hvor vi lagde demo versjoner av appen - for å lære oss teknologien. I fase to er det SignalR som er nytt for oss. Denne typen sanntidsteknologi har vi ikke brukt før, så det er en del å lære her også.

En annen unik utfordring er at gruppen er splittet på to ulike versjoner av appene: IOS og Android. De som lager Android versjonen er ikke med på IOS utviklingen, og motsatt. Men likevel må de to gruppene samarbeide for å sørge for at appene ble så like som mulig.

Begge disse utfordringene blir fint håndtert av den kontinuerlige kodeoppdateringen vi har - gjennom pull requests, kodegjennomganger og statusmøter."

Spørsmål: "Hvordan har avgrensingene til prosjektet påvirket valg av utviklingsmetodikk?"

Dette gjelder kun for dette prosjektet. Avgrensninger = valg dere har tatt (feks. scope på prosjekt, funksjonalitet, tid)

Svar: "Det har vært en del endringer som har påvirket utviklingen av dette prosjektet. Først og fremst har en nøkkelperson har nylig sluttet i banken, og en del ansvar har fordelt seg på andre utviklere. Dette betyr at det måtte gjøres en del omprioriteringer og endring på oppsatt prosjekttid.

Av den grunn har vi ikke tenkt på å endre metodikken. Det hadde blitt for kaotisk å endre utviklingsmetodikk midt opp i dette. Men, som jeg nevnte i det første svaret, så endres stegene i metodikken hele tiden. Så enskal ikke se vekk fra at en del av DevOps metodikken endres når vi kommer lengre ut i prosjektets levetid. Men dette blir da mindre endringer."