Forprosjekt - OneCall Metrics

Presentasjon av gruppe 33

Studentene:

Elzat Arkin	s354390	*Informasjonsteknologi*
Aksel Holm Jensen	s314807	*Anvendt Datateknologi*
Jenny Liang Nergård	s312980	*Anvendt Datateknologi* ***(talsperson)***
Jonas Nicolaysen	s354501	*Anvendt Datateknologi*
Aleksandar Perendic	s351906	*Dataingeniør*

Veileder OsloMet:

Boning Feng, boning.feng@oslomet.no, Tlf. kontor: (+47) 67 23 87 14, mob: (+47) 958 97 687.

Presentasjon av oppdragsgiver

Kontaktperson hos OC:

Tonje Moe Smith-Hansen, *Application Manager*
Tlf.: (+47) 450 02 068
Mail: tonje.smith-hansen@telia.no

Oppdragsgiver:

OneCall (OC) - Telia Company
Adr.: (Økern Portal) Lørenfaret 1,
0580 Oslo

Veiledere hos OC:

Ole-Petter Wikene (server)
Boyd Hermansen (systemutvikler)
Cato Berglie (delivery manager)
Arild B Nilsen (databaser)

Sammendrag

Vår gruppe skal utføre oppdrag hos telekom-selskapet OneCall. Her skal vi utvikle en applikasjonsløsning som kan fremvise demografi over deres kundebase, statistikk over enkeltkunder og deres bruksmønster og bruke maskinlæring for å vise en sannsynlighet for om kunden er i risiko for å avslutte kundeforholdet. Vi har mye frihet til å løse oppgaven slik vi selv ønsker, og vil derfor anvende teknologi vi selv har god kjennskap og erfaring med. Vi kommer til å få god nytte fra erfaringen vi har hatt fra fagene i studieløpet, og vil lage en fullstendig løsning som er utfordrende, relevant for arbeidslivet og innenfor tidsrammen.

Dagens situasjon

OneCall (OC) er et telekom-selskap eid av Telia, men opererer som en separat enhet. OC har alltid markedsført seg som det folkelige mobilselskapet og ønsker gjøre mobiltelefoni enklere og smartere, for folk flest. OC har mye data om sine kunder som de mangler god statistikk på. De ønsker å lære å kjenne kundene sine bedre slik at de kan forbedre og tilpasse tjenester for dem.

"Churn" er et begrep som handler om at eksisterende kunder avslutter kundeforholdet sitt og bytter til en annen leverandør. Årsakene til dette er mangefasettert og uklar. OC er spesielt opptatt av å motvirke churn som utgjør et stort økonomisk tap for dem. Derfor ønsker de at vi skal lage en løsning for dem som kan være behjelpelig til å få bedre oversikt over deres kunder og forhåpentligvis finne årsaker og sammenheng til churn. I tillegg til å fremvise enkel statistikk og oversikt over deres bruk slik at abonnementspakker enklere kan skreddersys for hver kunde.

Vår gruppe har fått to store tabeller med data. Den ene har anslagsvis 15 millioner rader med CDR (Call Data Records), mens den andre tabellen inneholder informasjon over nesten 30 tusen kunder og deres profil. Vår oppgave er å lage en applikasjon for oppdragsgiver som visualiserer og presenterer denne dataen på en måte som gir innsikt om kundenes bruksmønster og churn-risiko. Applikasjonen skal både kunne vise statistikk slik at abonnementer og tilbud kan bedre skreddersys, men også anvende kunstig intelligens/maskinlæring til å oppdage uante trender til hvorfor noen velger å avslutte kundeforholdet og lignende. Applikasjonen skal kun anvendes av oppdragsgiver til internt bruk.

Gjennom smidig utvikling skal vi bygge hele applikasjonen fra backend til frontend mot et ferdig produkt, med design, utvikling, testing og implementering. Vi får disponert deres lokaler to faste dager i uken, med mulighet for flere dager om vi har behov for det. I tillegg har vi fått tildelt flere veiledere fra ulike arbeidsområder, slik at vi best mulig kan få nytte av deres kunnskap. OC vil bistå økonomisk ved behov for ytterligere rammeverk og andre software-avhengigheter som vi studenter ikke har til disposisjon via universitetet, og som ikke lar seg skaffe kostnadsfritt.

Problemstilling

Vår opprinnelige problemstilling var: "Hvordan kan vi på best mulig måte fremvise data om eksisterende kunder som kan gi nyttig innsikt for oppdragsgiver?". Denne problemstillingen er mer rettet mot frontend og visualisering, mens vårt mål er mer rettet mot backend og bygging av en

applikasjon som kan bruke maskinlæring til å hente ut nøkkelinformasjon. Etter flere møter og samtaler med oppdragsgiver utarbeidet vi en problemstilling bedre rettet mot vår målsetting;

Hvordan konstruere en løsning som vil gi OneCall mer innsikt i kundene sine, også med hensyn til churn risiko?

Mål og rammebetingelser

Vi opplever at oppdraget vi er blitt utdelt er omfattende og utfordrende, men oppnåelig innenfor både læringsmål, vanskelighetsgrad og tidsrammene. Det aller meste vi skal jobbe med dekker flere aspekter av fag vi har hatt undervisning i. Vi vil kunne få utnyttet lærdom og pensum fra fagene Programmering, Databaser, Webprogrammering, Systemutvikling, Operativsystemer, Testing av programvare, Datasikkerhet, Webapplikasjoner og Introduksjon til Kunstig Intelligens. Vi kommer trolig også til å sneie innom Webutvikling, Webdesign samt Visualisering og MMI. Vår kjennskap til de nevnte fagene og erfaring med verktøyene vi skal bruke gjør at vår målsettinger og krav skal være mulig å fullføre. Da det også er ønskelig med anvendelse av maskinlæring/kunstig intelligens som vi har kun har begrenset erfaring med mener vi at oppdragets vanskelighetsgrad ligger rundt middels/høy.

Vi har delt målene inn i 4 milepæler som igjen er delt inn i mindre delmål. De første 3 er ufravikelige og helt essensielle for å kunne omtale oppgaven som fullført. Den siste er ønskelig for å kunne forbedre løsningen og gjøre den mer anvendbar for videre bruk etter endt bacheloroppgave.

Milepæl 1: Konstruer en MVP

- Hent og fremvis en kunde.
- Utvikle funksjonelt dashboard som fremviser demografi og statistikk.

Milepæl 2: Implementere sikkerhet.

- Innlogging med hashing og salting.
- Bruk av f.eks. tokens som JWT, eller lignende som egen mikrotjeneste.
- Fungerende brannmur.

Milepæl 3: Statistisk analyse av kundene

- Oppdag trendene og bruksmønsteret til kunder som er i risikosonen for churn.
- Presentere en sannsynlighet for om kunden befinner seg utenfor eller innenfor churn risiko .
- Tilretteleggelse for ulike type søk med bruk av Python og ML.
- Kjøre innsikt fra maskinlæring på en egen webserver med et API som kan nås av hoved applikasjonen.

Milepæl 4: overføre til mikrotjenester

- Bruke Container og VM til å gjøre applikasjonen om til en mikrotjeneste basert løsning som legger til rette for mer skalerbarhet og fleksibilitet i løsningen.

Krav

OC sitt krav er at vi skal kunne presentere en løsning som de kan benytte i henhold til deres ønsker. Forutenom det har de gitt oss mye frihet til å spesifisere krav, samt de verktøy vi er mest komfortable med. Ut fra samtalene vi har hatt med dem har vi utarbeidet krav og use-caser:

Funksjonelle krav

1. Løsningen skal kunne presentere ulike utsnitt av dataen
2. Løsningen skal kunne gjøre kall til databasen og tilgjengeliggjøre disse gjennom API'er/endepunkter
3. Løsningen skal kunne kategorisere kundene i ulike kategorier basert på churn risiko.
4. De ansatte skal kunne logge seg inn på løsningen gjennom credentials, og opprettholde tilkoblingen med sessions (ikke avklart om det skal være bruk av SAML/intern Azure kobling)

Ikke-Funksjonelle krav

1. Løsningen skal være oversiktlig og brukervennlig
2. Løsningen skal ha et rent og pent design uten unødvendige forstyrrelser.
3. Løsningen skal være responsiv/universelt utformet.
4. Løsningen skal ha god responstid/være rask.

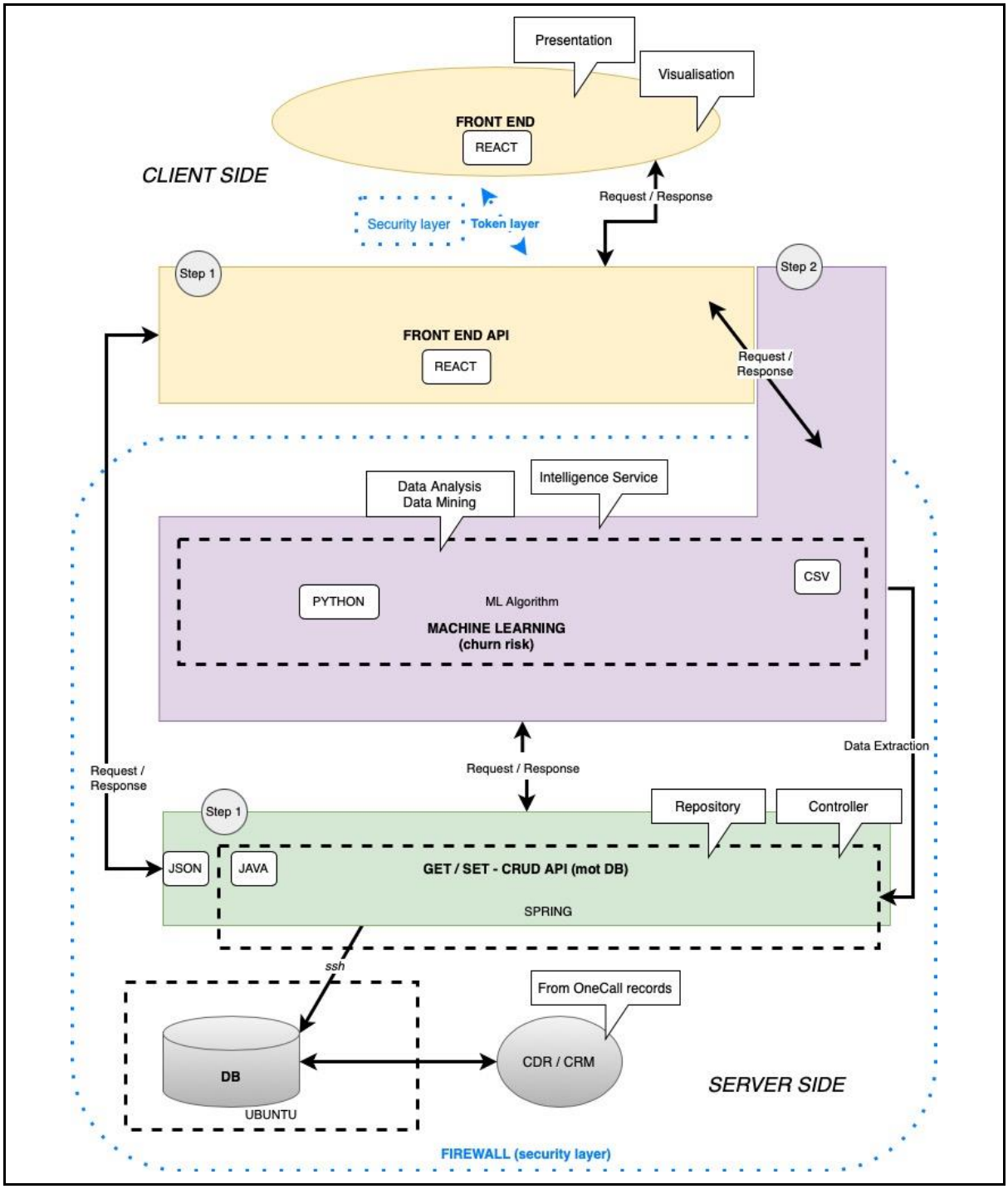
Use-caser

1. Som ansatt skal man kunne nå applikasjonen og data.
2. Som ansatt skal man kunne bli presentert detaljert statistikk for en eller flere kunder.
3. Som ansatt skal man kunne bli presentert overordnet statistikk over kundebasen
4. Som ansatt skal man kunne se sannsynlighet for hvorvidt en kunde er i risiko for å avslutte kundeforholdet.
5. Som ansatt skal kan man kunne bytte ut datakilden/rådataene og fortsatt ha en fungerende løsning.

Løsninger og alternativer

Plan for løsningsarkitektur

Løsningsarkitekturen (på det nåværende tidspunkt) er illustrert i skissen under. Systemet er komponert i form av flere ulike lag med forskjellige oppgaver. Et frontend-lag basert på React som skal brukes til å forme brukergrensesnitt hvor data blir hentet inn og fremvist. Requests fra frontend blir sendt videre til API-laget i Spring som inneholder Getters og Setters for CRUD-funksjonaliteter som videre henter og oppdaterer data i databasen. Databasen kjører i en egen Ubuntu-server. Det er i tillegg et Python-lag med maskinlæring som skal brukes til churn analyse. Python-laget er koblet til API-laget som presenterer data for churn analyse.



Research og konsept

Det aller første vi måtte gjøre er å bli kjent med selskapet (oppdragsgiver), produkter de tilbyr, måten de jobber på, ansatte og hva slags applikasjon de trengte internt. Det viste seg at OneCall, som mange andre selskap, sitter på mye rådata som trenger å bearbeides for å ekstraktere nyttig informasjon. Etter at vi ble kjent med oppgaven begynte vi å undersøke nærmere hva det innebærer for oss, lese artikler om telekombransjen og ulike måter lignende selskap "fanger" og forsøker å beholde sine kunder. Sistnevnte er et betraktelig fokusområde da det omfatter churn som vi er nokså uerfarne med. Tilstrekkelig research her vil også gi innsikt i maskinlærings-modeller vi kan bruke for å best finne churn for hver kunde i vår løsning. Videre ble det viktig for oss å få innsikt i

data vi skal jobbe med, og hva vi kan få ut av det. Dette gjorde at vi kunne bli enige om hvilken teknologi som skal benyttes og sammen med oppdragsgiver landet vi på det som er mest nyttig, både for oss som team og for OC som selskap.

Design/UX

Når det gjelder selve prosjektet og design ligger fokus på å utføre oppgaven riktig og levere nyttige data fremfor å bruke tid på å designe hele løsningen selv. Derfor akter vi å bruke en React-template for dashboard som vi tilpasser etter våre behov. Vi vil også se på noen alternativer som innebærer bruk av Googles Material Design. Dette kan endre seg underveis, men er det vi sikter mot på det nåværende tidspunkt.

Utvikling

Vi ønsker å følge flere av konseptene vi lærte i løp av studieperioden, som også brukes i arbeidslivet. Dette gjelder f.eks. fordeling av prosjektet/koding i flere sprints, og videre fordeling av små oppgaver i teamet basert på Jira tickets. Vi har valgt å legge opp arbeidet etter Scrum. Denne arbeidsprosessen anvender roller som en Scrum master og har definerte arbeidsoppgaver tildelt en spesifikk person som skal løses på en tidsperiode kalt Sprint. Endringer kan gjøres etter sprinten er ferdig, og antall story points kan brukes til å måle produktivitet. Vi har bestemt oss for en "Scrum master" som holder styr på planen og passer på at den følges, samt fordeler oppgaver basert på "rolle" i teamet. På den måten får alle være med på oppgaven likt, og dessuten kunne bidra overalt det er nødvendig, altså forarbeider, utvikling, testing, dokumentering og rapportering. Underveis vil vi ha hyppige møter med oppdragsgiver for å forsikre oss om at vi er på god vei mot noe de er fornøyd med.

Lansering

Lansering er ikke avklart med vår oppdragsgiver, men vi forventer at løsningen blir klar, og kan tas i bruk før vi presenterer selve prosjektet for fakultetet. Vi antar at vi har en enkel MVP på plass i løpet av kort tid, og at vi fortsetter å bygge på den frem til ferdigstillelse i midten av mai. Parallelt utvikler vi tester og skriver dokumentasjon rundt utviklingsprosessen.

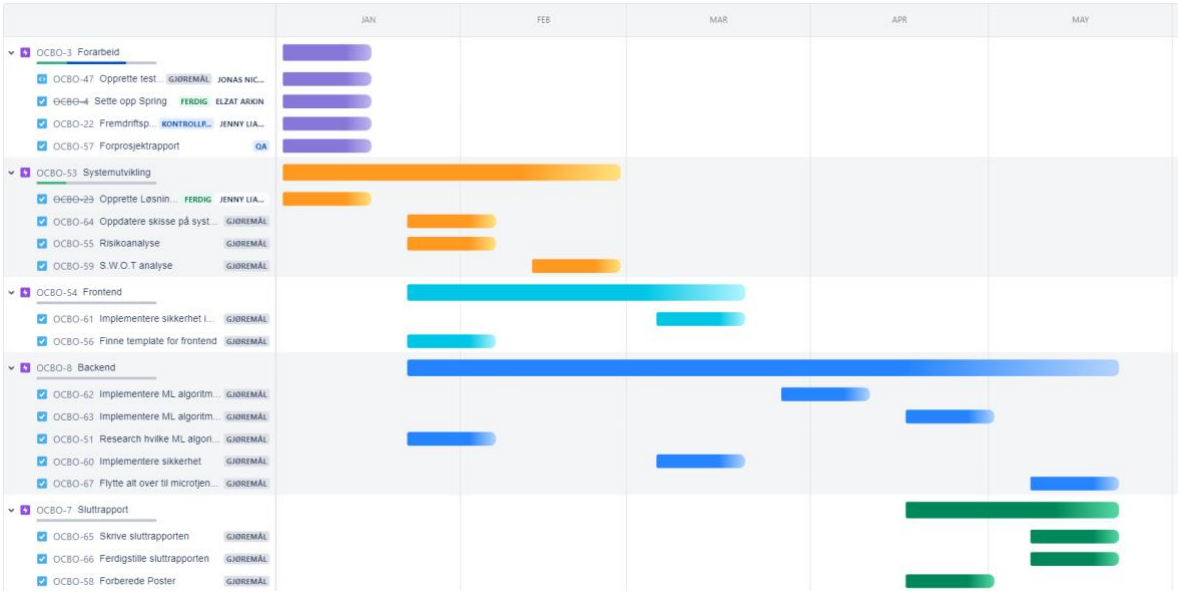
Teknologi og verktøy

Valgt teknologi	Alternativ teknologi	Årsak / analyse av virkninger
Backend: Java/Spring Boot, MySQL server, Ubuntu VM(hosting), Python(Maskinlæring)		
Spring Boot / Java	.Net / C#	Vi gikk for Java med Spring Boot siden vi har mer erfaring med dette rammeverket og språket, det er brukt i arbeidslivet, og er nyttig erfaring for oss.
MySQL	PostgreSQL	MySQL er også mye brukt i næringslivet og vi har også brukt denne i skolearbeid. I tillegg til det har OneCall valgt å utlevere data til oss i form av en MySQL tjener.
Ubuntu server	Lokal server	Data lagret på en server kan enkelt nås fra flere steder ved hjelp av en nettverksforbindelse, noe som gir flere

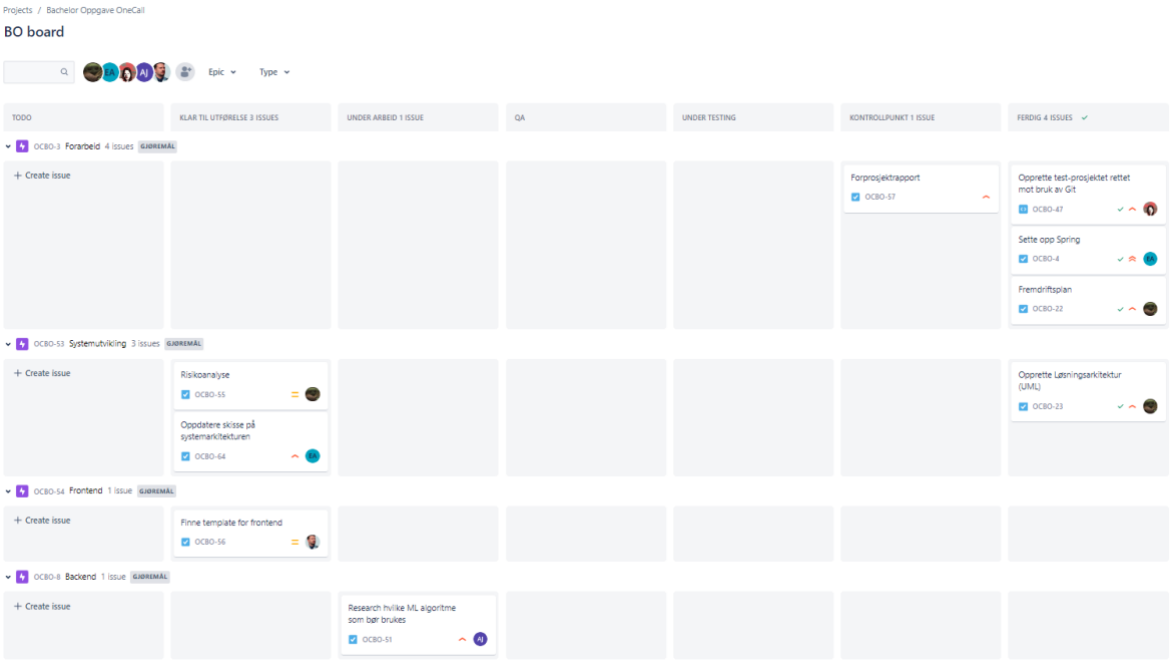
		brukere tilgang til dataene samtidig. En server kan lett oppgraderes med mer lagringsplass eller ytelse når det er nødvendig, mens lokal lagring kan være begrenset av fysiske begrensninger.
Python ML	Manuell analyse	Vi valgte å gå for AI fordi modeller kan analysere store mengder data mye raskere, og finne mønstre i data som kan være vanskelige eller umulig å oppdage manuelt. AI-analyser kan bidra til å forbedre beslutningene som tas i en virksomhet ved å gi mer presis og relevant informasjon enn manuell analyse kan gjøre. Det er nettopp det vi trenger for oppgaven vår.
Frontend: Javascript(React), HTML, CSS		
React	Angular	Vi valgte React siden dette er mest brukt i næringslivet og OneCall bruker dette selv. React er et svært fleksibelt rammeverk, har mye dokumentasjon på nettet og er kjent for stabilitet og ytelse.
Andre: Jira, GitHub, MySQL Workbench (Oracle), Jetbrains IntelliJ		
Jira	Trello	Jira er ledende og mye brukt i bransjen, og en programvare flere av oss er kjent med fra arbeidslivet. Telia bruker også Jira, men vi har opprettet vår egen side hvor vi har oversikt over tickets, hva vi jobber med og tildeling av oppgaver i teamet. Fordelen med å bruke Jira er å holde styr over store grupper.
GitHub Actions	GitLab CI/CD	Vi valgte å bruke GitHub Actions ettersom vi har brukt mest GitHub på skolen. Actions er basert på continuous integration and continuous delivery, som kan benyttes for deployment av applikasjonen.
JetBrains IntelliJ	Eclipse IDE	IntelliJ valgte vi fordi vi har brukt det mye i studieløpet, og fordi integrasjon til Spring er innebygd. Det forenkler en del av arbeidet, og gjør at vi kan heller fokusere på det tekniske.
Figma	Adobe XD	Figma er mer standardisert i markedet og blir brukt i større grad, dermed er det lettere å finne dokumentasjon på dette verktøyet. I tillegg tilbyr Figma gratis studentpakke som passer vår situasjon bedre.
Monolitt arkitektur	Mikrotjenester arkitektur	I utgangspunktet er koden som skrives den samme, så det som er forskjellen er fordelingen på ulike segmenter/containere. Mikrotjenester er fortsatt mulig, men oppgaven vil ikke være så omfattende at mikrotjenester er et nødvendig fokusområde.

Arbeids- og fremdriftsplan

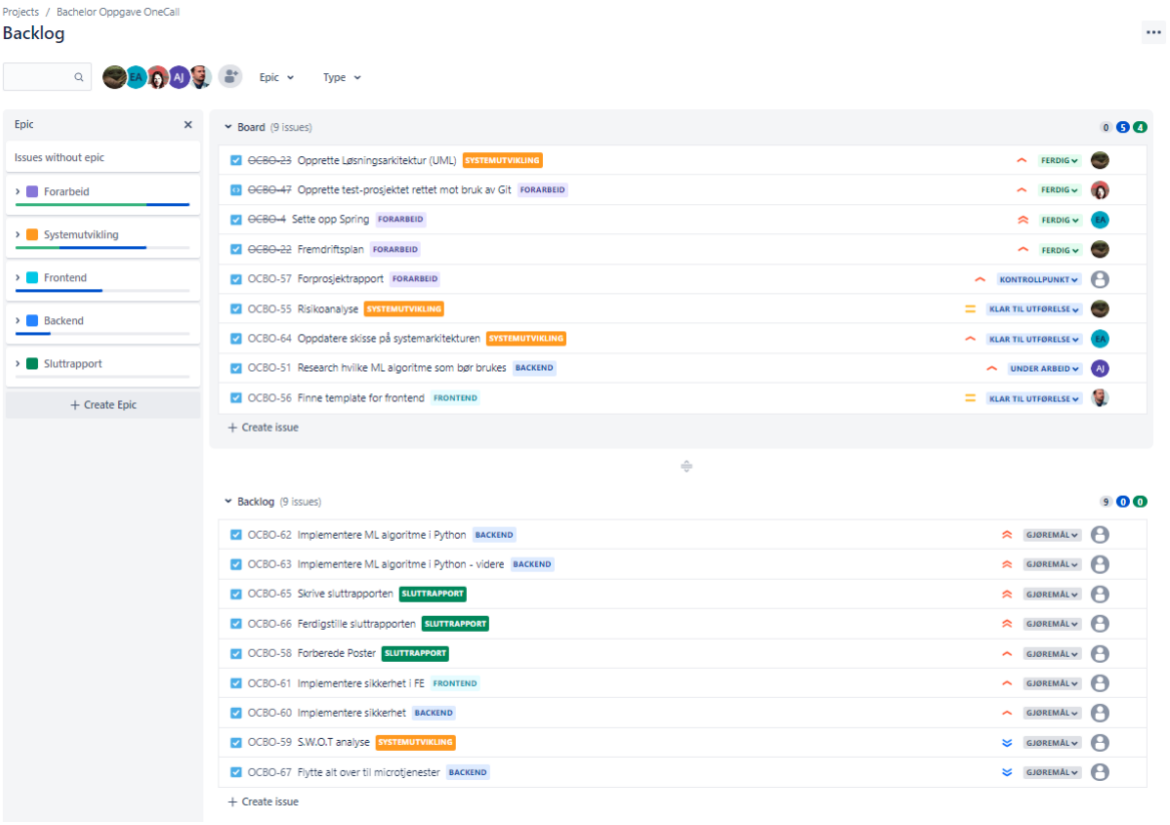
Vi har valgt å bruke Jira som prosjektstyringsplattform til å styre arbeidsflyten og holde oversikt over fremdriften. Jira er et skreddersydd verktøy for smidig utvikling som f.eks. Scrum, med ferdige løsninger som gjør prosjektstyringen enklere og mer oversiktlig for et team. Vi har en gratisversjon så vi har ikke full tilgang på alle funksjoner som f.eks. 'sprint'-feature, så vi har valgt å legge inn frister og emnefelt som avgrense sprintene.



Vi har fordelt oppgavene i 8 sprinter med 3 ukers intervaller per sprint. Dette vises i Jira sin fremdriftsplan-visning.



Scrum-tavlen viser oppgavene, hvem de er fordelt til og hvor i arbeidsprosessen de er.



I backlogen ser vi hvilke oppgaver som må gjøres for hele prosjektet. De som ligger i tavlen er de som skal gjennomføres i inneværende sprint.