Bacheloroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

Jenny Farstad Blindheimsvik
Linda Katrine Larsen
Nora Evensen Jansrud

Visualisering av kompetanse

Bacheloroppgave i Dataingeniør
Veileder: Elise Klæbo Vonstad
Mai 2022

NTNU
Norwegian University of
Science and Technology

Jenny Farstad Blindheimsvik
Linda Katrine Larsen
Nora Evensen Jansrud

Visualisering av kompetanse

Bacheloroppgave i Dataingeniør
Veileder: Elise Klæbo Vonstad
Mai 2022

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

NTNU
Kunnskap for en bedre verden

Sammendrag

Datavisualisering er et stort fagfelt med en lang historie. Det mest kjente verket innenfor datavisualisering er kanskje Edward R. Tuftes bok *The Visual Display of Quantitative Information.* Datavisualisering av kompetanse er et relativt snevert fagfelt, hvor det har blitt gjort lite forskning. Det finnes enkelte produkter på markedet som tar sikte på å visualisere kompetanse, men det har ikke blitt observert produkter som visualiserer kompetanse basert på CV-er. Det finnes heller ikke produkter som visualiserer kompetanse med formål om å sette sammen team.

I løpet av denne prosjektperioden har det blitt utviklet en løsning for å visualisere kompetanse basert på standardiserte CV-er, med formål å undersøke om et verktøy kan bidra til å fasilitere prosessen med å sette sammen profesjonelle team. For å finne ut av dette har det blitt utviklet en webapplikasjon og samlet inn data gjennom brukertester for å undersøke problemstillingen. Det har også blitt gjort en litteraturstudie av datavisualisering, samt lagt ned et arbeid i å kategorisere dataene.

Scrum har blitt brukt som utviklingsmetode i dette prosjektet. Applikasjonen har blitt utviklet med en lagdelt struktur, der Vue har blitt brukt til å skrive klienten, mens ASP.NET har blitt brukt som programmeringsspråk for serveren. For prosjektstyring og utvikling har Azure Devops med diverse funksjonalitet blitt benyttet.

Prosessen for teamsammensetning er i dag basert på prosjektleders nettverk og subjektive oppfatning av ansatte hvor CV-er brukes som et supplement. Ved å visualisere kompetansen CV-er inneholder kan en prosjektleder raskere få oversikt over større mengde ansatte. Resultatene fra brukertestene har gitt en indikasjon på at det finnes forbedringspotensial i prosessen med teamsammensetting, samt at produktet med noe videre implementasjon, kan være et nyttig verktøy for å fasilitere denne prosessen.

Abstract

Data visualization is a large field with a long history, and much has been accomplished within the field in recent times. The most famous work in the field of data visualization is perhaps Edward R. Tufte's book *The Visual Display of Quantitative Information*. Visualization of competence is a relatively narrow field, where little research has been done. There exist some products on the market that aim to visualize competence, however to our knowledge there are not any that visualizes competence based on CVs. Moreover, there are no products that visualize competence for the purpose of creating teams.

During this project period, the group has developed a solution to visualize competence based on standardized CVs, with the aim of investigating whether such a tool can contribute to facilitating the process of assembling professional teams. To investigate this, a web application has been developed, and data has been collected through user tests. A literature study of data visualization has also been carried out, as well as categorizing data

Scrum has been used during the development process. The application has been developed with a layered structure, where Vue has been used to write the client side of the application, while ASP.NET has been used as the programming language for the server side. For project management and development, Azure DevOps with various functionality has been used.

The process of composing teams is today based on the project manager's network and subjective perception of employees where CVs are used as a supplement. By visualizing the competence contained by the CVs, a project manager can more quickly gain an overview of a larger number of employees. The results from the user tests have given an indication that there is a potential for improvement in the process of team composition, and that the product with some further implementation, can be a useful tool to facilitate this process.

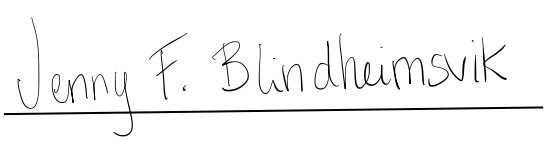
Forord

Denne oppgaven er skrevet i anledning av Bacheloroppgave IDATT2900 ved studiet Dataingeniør, NTNU, utført våren 2022. Oppgaven er stilt av Tietoevry Norway AS (heretter referert til som Tietoevry), og har tittelen *Visualisering av kompetanse.* Oppgaven ble gjennomført av Jenny Farstad Blindheimsvik, Linda Katrine Larsen og Nora Evensen Jansrud.

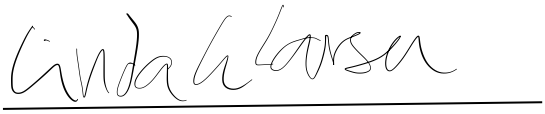
Valget av oppgave falt på *Visualisering av kompetanse* fordi gruppen mente oppgaven virket spennende. Det ble også ansett som en utfordrende, men overkommelig oppgave. Gruppen ønsket å skrive oppgave for en ekstern bedrift for å bli kjent med næringslivet. Samtidig var oppgaven av en slik art at den virket åpen nok til at gruppen kom til å få lov til å sette sitt eget preg på det ferdige resultatet.

Generelt kan arbeidet med oppgaven deles opp i fire hoveddeler; forarbeid, produktutvikling, administrativt arbeid, og avsluttende arbeid. Forarbeidsfasen, bestående av oppstart- og planleggingsarbeidet, omfatter oppstartsmøter, utvikling av forprosjektplan og andre dokumenter som hører til planleggingsfasen. Produktutviklingsfasen består av utviklingen av produktet samt brukertester og blir gjennomført iterativt gjennom metodikken Scrum. Det administrative arbeidet består av arbeid som vil foregå under hele prosjektet, som møter og utarbeiding av hovedrapport og systemdokumentasjon. I slutten av prosjektperioden omfatter det avsluttende arbeidet ferdigstilling av all dokumentasjon samt utarbeiding av denne hovedrapporten.

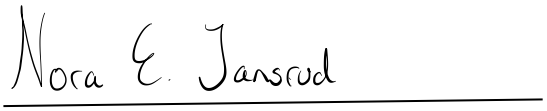
Under arbeidet med oppgaven har gruppen fått nytte av god og kvalifisert hjelp fra mange hold. En stor takk utstedes til veileder Elise Vonstad Klæbo, som har gitt verdifull tilbakemelding underveis i hele prosjektet. Vi ønsker å takke Tietoevry for en spennende og utfordrende oppgave, samt kontorplass og tilgang på nødvendig utstyr. En takk utstedes også til oppdragsgiver og produkteier Roar Gjøvaag, som har gitt mye god tilbakemelding på oppgaven underveis, og som også har hatt en viktig rolle i forståelsen av hvordan produktet skulle ende opp. En takk utstedes til Stian Andreas Solheim som har stilt opp og bidratt med rådgivning og hjelp på den tekniske biten av oppgaven. Det utstedes også en takk til alle som har stilt opp på og hjulpet til med brukertester. Takksigelser utstedes også de som har hjulpet til med å lese igjennom hovedrapport og bidratt med faglige råd gjennom prosessen.



Jenny Farstad Blindheimsvik



Linda Katrine Larsen



Nora Evensen Jansrud

Oppgavetekst

Den opprinnelige oppgaveteksten er gjengitt i avsnittet under, med enkelte språklige tilpasninger. Ved slutten av prosjektet lå det samme grunnlaget for oppgaven. Prioriteringer, innsnevringer og tilpasninger har blitt gjort underveis i prosjektet ettersom man møtte på utfordringer eller fant dette hensiktsmessig for prosjektfremdriften.

Oppgaveteksten utstedt var som følger:

I komposisjoner av team til prosjekter er det ofte vanskelig å få god oversikt over kompetanse og erfaring teamet innehar som gruppe eller individuelt. Dette skaper en utfordring for interessenter som skal planlegge og sette sammen team fordi det kan være tidkrevende å manuelt kartlegge metoder, verktøy og teknologier som teamet har erfaring med. Dette er viktig fordi man reduserer risiko og øker sannsynligheten for en suksess i et prosjekt om man på en effektiv måte kan vurdere om teamet har en god nok komposisjon av medlemmer.

- *Løsningen skal visualisere kompetanse til en eller flere personer i form av diagrammer, grafer eller andre relevante framstillinger av innholdet.*
- *Løsningen skal bruke input fra tekstdokumenter i form av PDF eller Word-fil som inneholder CV.*
- *I løsningen skal man kunne sette sammen individers profiler i team.*
- *Løsningen skal kunne sammenligne forskjellige profiler og team-sammensetninger.*
- *Løsningen skal kunne eksportere visualiseringen i form av vektorgrafikk.*

I oppstarten av prosjektet blir det etablert at datagrunnlaget som skal benyttes er oppdragsgivers standardiserte CV som alle ansatte har gjennom tjenesten CV-partner. CV-partners tjenester fokuserer på å bruke CV-er til å bygge tilbud i sammenheng med salg (*CV Partner - For Bids and Proposals*, u.d.).

En mer detaljert beskrivelse av kravene og visjonen for oppgaven kan leses i vedlegg 2, Visjonsdokument og vedlegg 3 Kravdokumentasjon.

Innholdsfortegnelse

Figurliste

Tabell-liste

Akronymer, forkortelser og begrepsforklaringer

API Står for *Application Programming Interface*, og er et hjelpeverktøy som brukes til å gi en definisjon på hvordan en utenforstående programmerer skal tilføye funksjonalitet eller tjenester til en applikasjon (Rossen, 2020).

DevOps Satt sammen av det engelske ordet for utvikler (developer) og det engelske ordet operations. DevOps kan best forklares med mennesker som jobber sammen for å bygge og levere sikkert programvare (*What is DevOps?*, u.d.).

DI *Dependency Injection* forklart i kapittel 2.6.2 Avhengighetsinjeksjon.

DSR *Design Science Research* forklart i kapittel 3.1 Forskningsmetode.

GDPR *General data protection regulation,* på norsk personvernforordningen, er en lov vedtatt av EU og er en del av Norges personopplysningslov (*Personvernprinsippene*, u.d.).

FURPS *Functionality, Usability, Reliability, Performace, Security.* Brukes for å klassifisere attributter for programvarekvalitet (‘FURPS’, 2021).

HTTP *Hypertext Transfer Protocol,* protokoll for å overføre HTML dokumenter mellom tjenere og klienter på internett (Dvergsdal, 2021).

HTTPS *Hypertext Transfer Protocol Secure* er en kommunikasjonsprotokoll som støtter sikker kommunikasjon over internett (Bartnes and Nätt, 2022).

IoC *Inversion of Control,* Invertering av kontroll, forklart i kapittel 2.6.2 Avhengighetsinjeksjon.

Lo-fi prototyper En form for prototyping som fokuserer på funksjonalitet, og ikke visuell utforming av produktet (Babich, u.d).

OSWAP *The Open Web Application Security Project* er en organisasjon som jobber for å forbedre sikkerhet på internett (*OWASP Foundation, Open Source Foundation for Application Security*, u.d.).

UI *User Interface,* brukergrensesnitt på norsk.

VCS *Version Control Systems.* Systemer for versjonskontroll.

WCAG *Web Content Accessibility Guidelines.* Retningslinjer for hvordan nettsteder og brukergrensesnitt utformes for å gjøre dem tilgjengelige uavhengig av brukerens funksjonsevne (*WCAG 2.0-standarden, Tilsynet for universell utforming av ikt*, u.d.).

W3C *World Wide Web Consortium* er en organisasjon som har som hovedmål å utvikle protokoller og standarder for teknologier som brukes på *World Wide Web (W3C, u.d.)*.

Kapittel 1 Introduksjon og relevans

1.1 Relevans

Oppgaven er stilt av Tietoevry, og har bakgrunn i deres ønske og behov for å visualisere kompetanse. Dette er også et behov som dukker opp på tvers av bedrifter og bransjer hvor fellesbetegnelsen er salg av kompetanse, gjerne i sammenheng med prosjektbasert arbeid i team. For å en god forståelse av relevansen til denne oppgaven, tas det utgangspunkt hvem Tietoevry er, samt deres arbeid og ønsker.

Tietoevry er et konsulentselskap som tilbyr IT- og produktutviklingstjenester (Tietoevry.com, u.d). De selger kompetansen til sine ansatte som enkeltindivider inn i eksisterende prosjekter, eller samlet i team til frittstående prosjekt hos kunden.

Ved innsalg av en slik ressurs benytter Tietoevry standardiserte CV-er for å kommunisere ressursenes kompetansenivå. Tietoevry ønsker å visualisere kompetansen representert i disse CV-ene på en bedre måte enn det som gjøres i dag. Bakgrunnen til dette kan sees i sammenheng med oppgavens effektmål.

Gruppen har definert fem effektmål for løsningen og ligger vedlagt i forprosjektplanen. Av disse har Tietoevry satt tre:

1. Produktet skal gjøre det tydeligere for leverandør og kunde hvilken kompetanse individ eller team innehar.
2. Produktet skal føre til mer effektive beslutninger rundt teamsammensetting.
3. Produktet skal føre til mer objektive beslutninger rundt teamsammensetting.

Målene er satt ut fra bedriftens ønsker og behov. Første mål står i direkte sammenheng med at det er ønskelig for oppdragsgiver å lettere selge team og å gi kunden et godt bilde av hvilken kompetanse de selger.

Andre mål har bakgrunn i et ønske om å effektivisere prosessen med å sette sammen team, særlig for prosjektledere med ansvar for en større mengde ansatte.

Siste mål handler om ønsket bedriften har om å sette sammen gode team som har den kompetansen det krever å løse oppgaven.

1.2 Problemstilling

Problemstillingen er definert av studentene under arbeidet med oppgaven. Problemstillingen ble basert på oppgaveteksten og effektmålene, og er spesielt knyttet til effektmål 2 og 3 definert i kapittel 1.1 Relevans. Oppgaven har som mål å besvare denne problemstillingen. Den ble formulert som følgende:

> «Hvordan kan en webapplikasjon for visualisering av kompetanse utvikles for å fasilitere sammensetning av et profesjonelt team?»

Dette innebærer å se på hvordan prosessen med å sette sammen team kan forbedres ved å bruke visualiseringer av kompetanse. For å fasilitere sammensetningen av profesjonelle team, ble problemstillingen delt opp i to; objektivisere prosessen og effektivisere prosessen. Dette er tett relatert til effektmålene spesifisert i kapittel 1.1.

For å effektivisere prosessen ble det fokusert på å lette arbeidsmengden rundt å sette sammen team. For å klare dette må prosessen rundt teamsammensetning først kartlegges i grove trekk. Deretter må det identifiseres områder hvor det er uforholdsmessig høy arbeidsmengde, slik at det kan foreslås en løsning på disse problemområdene. Et annet viktig aspekt av å effektivisere prosessen er å sørge for en god brukeropplevelse (Hartson and Pyla, 2012, s. 78).

Under arbeidet med visualisering, ble det utformet en hypotese som gikk ut på at den fortrukne visualiseringen ville avhenge av størrelsen på datasettet den baserer seg på.

1.3 Hovedrapportens struktur

Her følger en beskrivelse av rapportens struktur. Dette er ment for å gi en oversikt over innholdet i de forskjellige kapitlene. Rapporten består av en innledning med sammendrag, forord, oppgavetekst og innholdsfortegnelse, samt seks kapitler.

I kapittel 1 avklareres bakgrunnen for oppgaven. I dette kapitlet settes oppgaven i sammenheng med behovet bak oppgaven, problemstillingen som skal besvares, samt en beskrivelse av strukturen til rapporten og en oversikt over akronymer og forkortelser.

I kapittel 2 beskrives teori og relevant litteratur. Hensikten med dette kapitlet er å gi leseren en teoretisk bakgrunn og grunnlag til hovedrapporten. Kapittel 2 inneholder teori om kompetanse, datavisualisering, webapplikasjoner, smidig systemutvikling, brukertesting og kvalitetssikring av kode.

I kapittel 3 beskrives metodene brukt for å løse oppgaven. Dette omfatter forskningsmetoden, valg av teknologi, valg av utviklingsmetode og arbeids- og rollefordeling. Forskningsmetode beskriver bruk av metoden Design Science Research med brukertester og litteraturstudie. I valg av teknologi beskrives og begrunnes valg av teknologier under produktutviklingen. I valg av utviklingsmetode beskrives nærmere hvordan gruppen har benyttet Scrum som utviklingsmetode med verktøyet Azure Devops, samt metode for databehandling. Arbeids- og rollefordeling beskriver til slutt prosjektets fordeling av roller og arbeidsområder. Hensikten med kapittel 3 er å gi en forståelse av hvordan gruppen har jobbet for å løse oppgaven, samt å begrunne valg av metoder.

I kapittel 4 presenteres resultatene. Kapitlet er strukturert i tre underkapitler; vitenskapelige-, ingeniørfaglige- og administrative resultater. I vitenskapelige resultater presenteres alle funn som er gjort på bakgrunn av datainnsamling gjennom den valgte forskningsmetoden. De ingeniørfaglige resultatene er resultater som har kommet frem basert på funksjonelle og ikke-funksjonelle krav. De administrative resultatene handler om hvorvidt man fikk til prosessen slik den var planlagt, og innebærer blant annet timefordeling og planlegging av utviklingsprosess.

I kapittel 5 diskuteres resultatene. Kapittel 5 har samme inndeling som kapittel 4. Her diskuteres årsaken til at resultatene ble som de ble. Samtidig pekes det på styrker og svakheter ved resultatene. Det gjøres også en vurdering av prosessen gruppen har hatt gjennom arbeidet med prosjektet.

I kapittel 6 kommer konklusjonen. Her besvares problemstilling. Her ligger også anbefalinger til videre arbeid, samt en samfunnsanalyse av prosjektet.

Kapittel 2 Teori og relevant litteratur

For å danne et teoretisk grunnlag for rapporten, har det blitt gjengitt relevant teori som bør forstås for å kunne lese og vurdere hovedrapporten. Kompetanse og visualisering av data inneholder teori og definisjoner som danner et teoretisk grunnlag for utformingen av produktet. Det er også gjengitt teori om tekniske aspekter av oppgaven for å gi et grunnlag for begrunnelse av metode i kapittel 3. Teori om forskningsmetode og systemutvikling er også inkludert, da dette er relevant i en systemutviklingsoppgave.

2.1 Kompetanse

2.1.1 Kunnskap, ferdigheter og erfaring

I arbeidslivet står begrepet kompetanse sentralt. Generelt defineres kompetanse som tilstanden av å være kompetent (Merriam Webster, 2022). I dagligtale formuleres det gjerne som 'å være i stand til', 'å være dyktig' eller 'å kunne noe' (Kompetanse, Snl, 2018). Det finnes flere måter å være kompetent på. Det er vanlig å dele inn kompetanse inn i to hovedkategorier; eksplisitt og taus kompetanse (Thorsvik and Jacobsen, s. 349). Taus kompetanse er kompetanse som man sitter med, men som man ikke kan sette ord på (Thorsvik and Jacobsen, s. 349). Taus kompetanse er ofte erfaringsbasert i form av tillærte vaner eller evner, og læres enten ved å erfare og prøve ut selv, eller ved å observere andre. Eksplisitt kompetanse er kompetanse man kan sette ord på, og forklare til andre (Thorsvik and Jacobsen, s. 349).

I arbeidslivet sees gjerne kompetanse på som en kombinasjon av kunnskap og ferdigheter (Merriam Webster, 2022). Kunnskap er den viten og innsikten en person har innen et spesifikt område (Holmen, u.d.). Ferdigheter er evnen til å utføre en oppgave og opparbeides gjennom å utføre oppgaver i praksis (Cambridge Dictionary, u.d.). Kompetanse kan dermed defineres som evnen til å anvende sine kunnskaper og ferdigheter innenfor et spesifikt område.

En viktig faktor for å måle kompetanse er erfaring. Erfaring baserer seg på det man samlet har lært av å observere samt opplevd og gjennomført (Erfaring, Snl, u.d.). Det omfatter erfaring man har ervervet gjennoms sin tid i arbeidslivet. Det er spesielt viktig å måle erfaring for å kunne favne taus kompetanse, da dette kan være vanskelig å vise på andre måter (Thorsvik and Jacobsen, s. 349).

2.1.2 Kompetanse innenfor IT

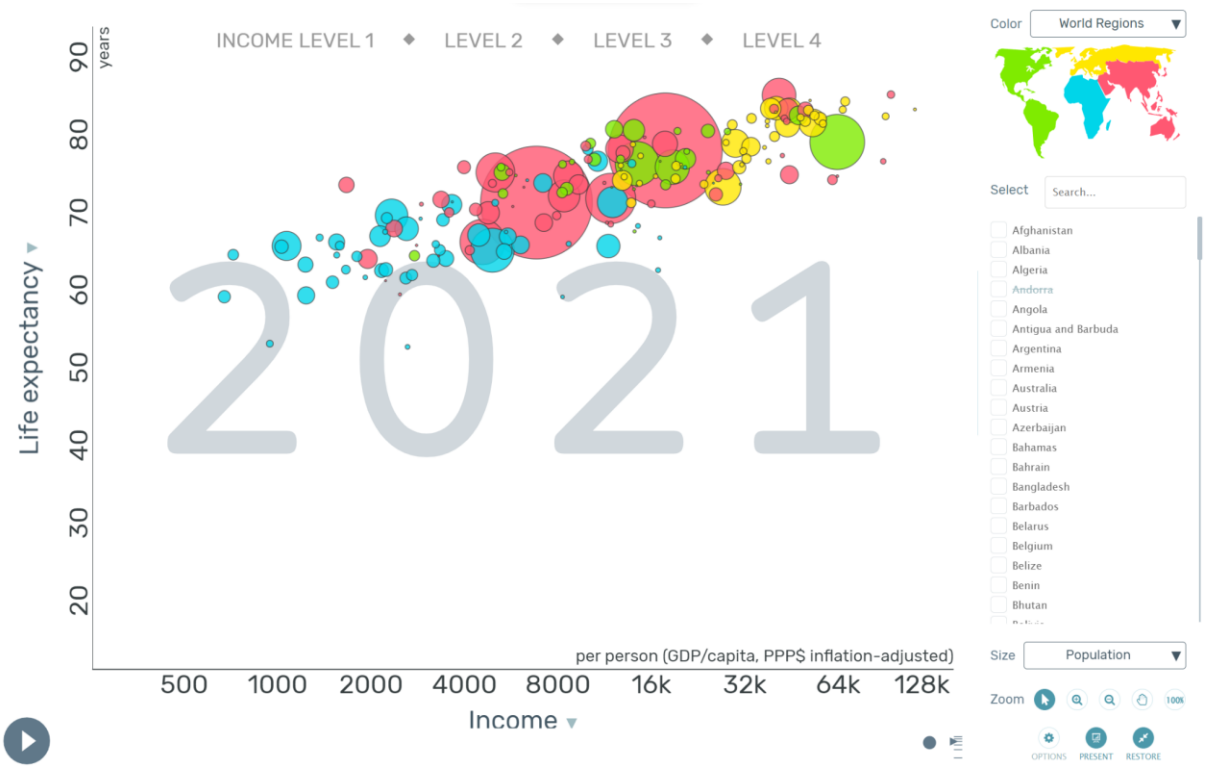
IT-industrien dekker alle bedrifter basert på innhenting, overføring, bearbeiding, lagring og presentasjon av informasjon (Spacey, 2022). De fleste industrier benytter seg i dag av informasjons- og kommunikasjonsteknologi og det finnes dermed et behov for ulik IT-kompetanse (Spacey, 2022). IT-konsulent firmaer fokuserer på å gi råd til organisasjoner/bedrifter om hvordan de best kan bruke informasjonsteknologi for å nå sine mål gjennom sine konsulenter (*IT Consulting*, u.d.). Siden behovene varierer fra industri til industri vil konsulentene ha forskjellige kompetanseområder som dekker ulike behov.

De siste tiårene har smidig systemutvikling revolusjonert informasjonsteknologi gjennom høyere suksessrate, forbedret kvalitet og redusert tid til marked. I prosjekter, som benytter smidig systemutvikling, settes det sammen tverrfaglige team da produktutviklingen har mange aspekter som krever forskjellig kompetanse (Rigby, Sutherland and Takeuchi, 2016). Hvilken kompetanse som trengs i et team, og hvor stort teamet skal være, vil variere ut fra størrelsen på prosjektet og hva prosjektet omhandler (West, u.d.).

2.2 Visualisering av data

2.2.1 Hva er datavisualisering?

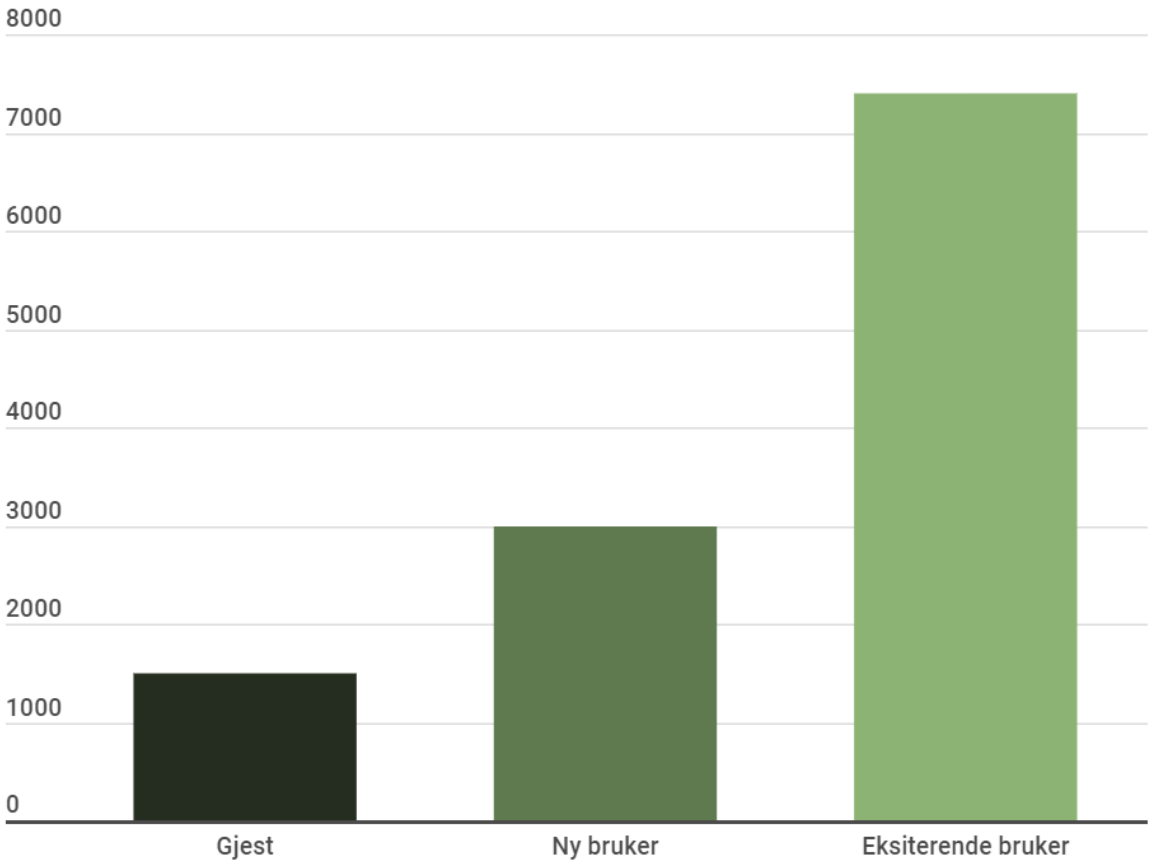
Visualisering av data er grafisk fremstilling av data (Unwin, 2020). Datavisualisering kan foregå på mange forskjellige måter, og med mange forskjellige målsetninger (Unwin, 2020). Eksempler på dette er scatter plots, se figur 2.1, som viser en oversikt over befolkningen i forskjellige land, søylediagram som viser oversikt over kjøp per brukertype, se figur 2.2, eller et kart som viser hvor mye koronasmitte det er i forskjellige kommuner i Norge.



Figur 2.1 Kombinasjon av Scatter plot og boblediagram.

Datavisualiseringens utforming kan variere utfra hvilken intensjon som ligger bak budskapet. Ønsker man å vise all data i et datasett vil man ofte bruke et annet diagram enn om visualiseringen kun skal fungere som en oversikt eller et sammendrag (Unwin, 2020).

Datavisualisering kan defineres som å omstille informasjon slik at den kan brukes i en visuell kontekst (Brush, 2020). Hovedpoenget med datavisualisering er å kunne skaffe et overblikk over store mengder data. Det gjør man for å lettere se mønster, trender og hovedlinjer i store datasett (Brush, 2020).

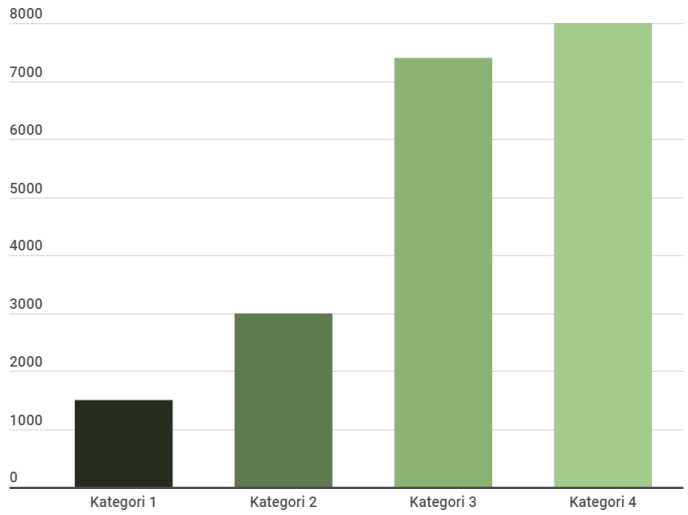


Figur 2.2 Søylediagram som illustrerer innkjøp per brukertype.

2.2.2 Relevante diagrammer

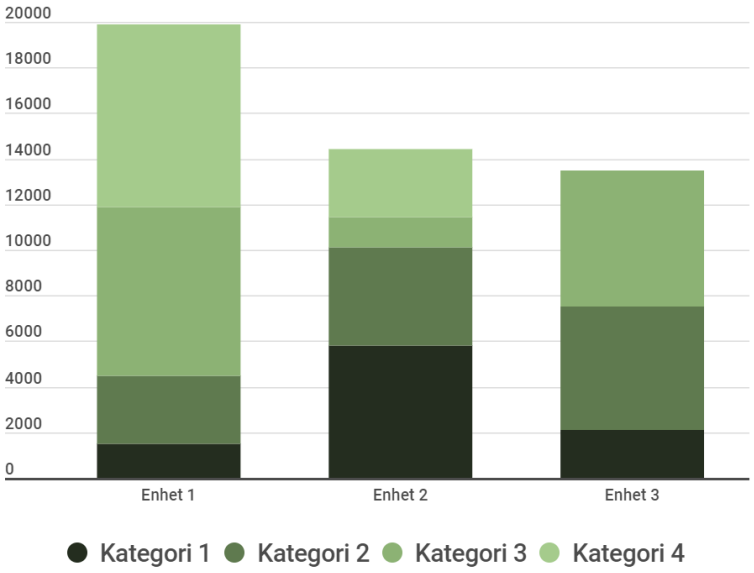
Dette er en oversikt over fem diagrammer som er relevante for å forstå hovedrapporten.

Søylediagram



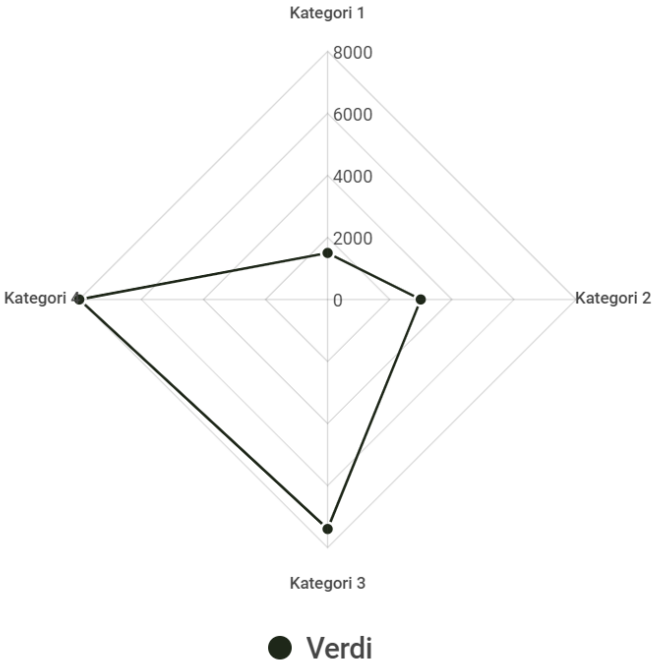
Figur 2.3 Søylediagram

Stablet søylediagram



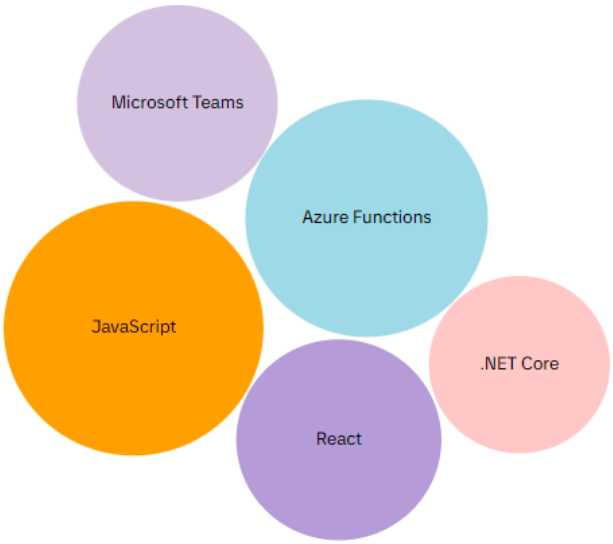
Figur 2.4 Stablet søylediagram

Radardiagram



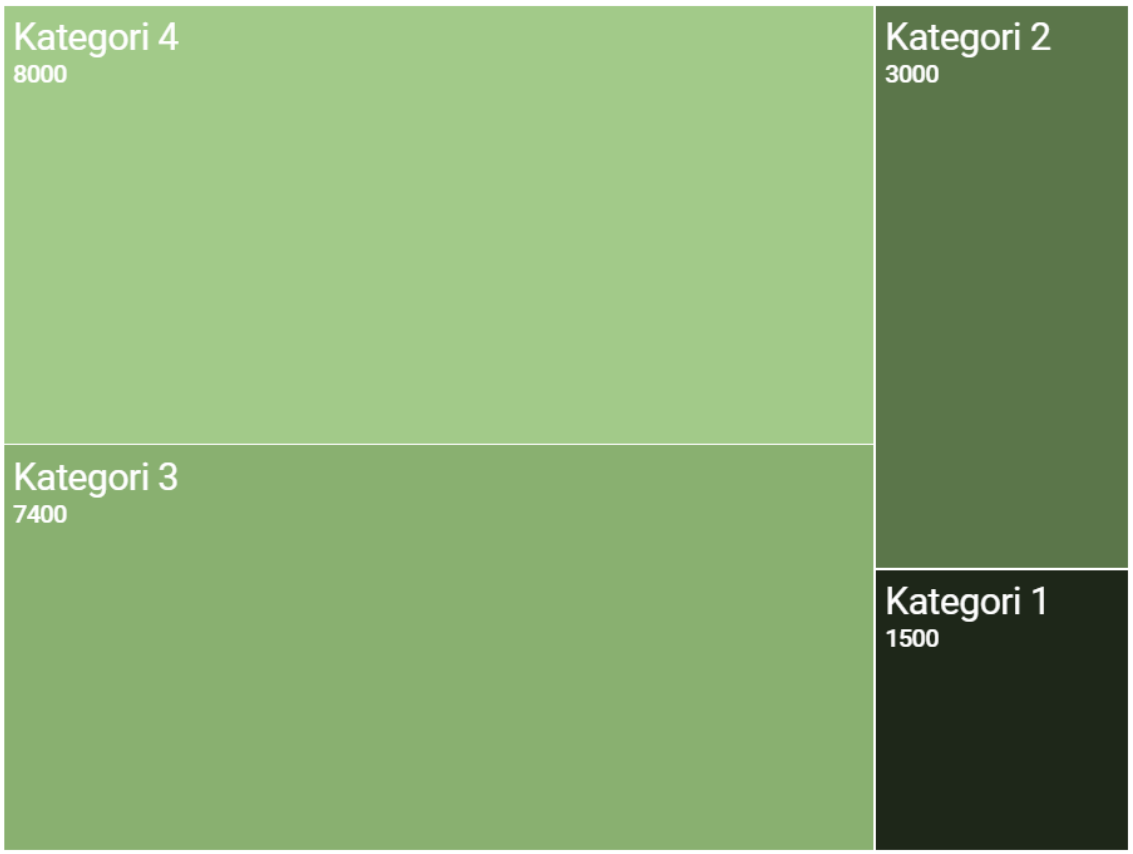
Figur 2.5 Radardiagram

Boblediagram



Figur 2.6 Boblediagram

Treemap



Figur 2.7 Treemap

2.2.3 Hvordan bør man visualisere data?

Visualisering av data kan være vanskelig, og gode visualiseringer krever kunnskap om design, samt hvordan mennesker oppfatter og prosesserer informasjon. Det eksisterer mange ulike versjoner av definerte grunnleggende retningslinjer for hvordan man skal oppnå gode visualiseringer. Det er dermed vanskelig å gi noen helt konkret metodikk for hvordan data bør visualiseres.

Edward R. Tufte skriver i sin bok *The Visual Display of Quantative Information* at statistisk grafikk består av komplekse ideer kommunisert med klarhet, presisjon og effektivitet (Tufte, 1983, s.13). Videre redegjør han for en rekke krav som bør innfris for å oppnå dette. Grafiske fremstillinger bør vise dataene. Den bør sørge for at den som observerer fremstillingen tenker på innholdet, ikke utforming. Fremstillingen skal unngå å forvrenge dataene eller endre på innholdet. Den bør også være godt leselig. For å oppnå dette skal man ikke vise mange nummer på små områder, og store datasett bør være sammenhengende. Fremstillingen bør oppfordre den som ser på til å sammenlikne forskjellige deler av dataene, og dele opp dataene i flere detaljnivåer (Tufte, 1983, s.13).

Jenny V. Freeman og Steven A. Julious (u.d) definerer seks retningslinjer for grafisk fremstilling av data. Det første prinsippet er at mengden informasjon bør maksimeres

med minimal mengde blekk. Det betyr at man skal vise så mye informasjon som mulig med så lite skrift og illustrasjon som mulig. Gode visualiseringer har ofte følgende egenskaper: et klart budskap, enkelt design, tydelige beskrivelser og integritet i intensjoner. En fremstilling bør ha en forklarende tittel, slik at det umiddelbart er åpenbart hva som fremstilles. Aksene bør også ha navn, slik at det er enkelt å oppfatte det de representerer. Bruken av rutenett, og andre liknende konstruksjoner bør holdes til et minimum, da de ofte kan virke forstyrrende på fremstillingen. Freeman og Julious skriver også at diagrammer i 3D-grafikk bør unngås, da dette kan være vanskelig å lese.

2.2.4 Dimensjoner i data

Man må ta hensyn til hvor mye detaljer det er ønskelig å vise frem når man skal visualisere data. Et datasett har en dimensjon (Stephanie, 2016). En dimensjon kan enkelt defineres som hvor mange søyler man bruker når man fremstiller dataene i en tabell.

For denne oppgaven er datasett med én- (1D), to- (2D) eller tre dimensjoner (3D) relevant. Det er mulig å visualisere data med flere enn 2 dimensjoner uten bruk av 3D-grafikk (Morgan, 2019). Et godt eksempel på en graf hvor data i 3D blir representert i 2D-grafikk er figur 2.1. Dimensjonene er størrelse på befolkning (representert i størrelsen på sirkelen), GDP (x-aksen) og forventet levealder (y-aksen).

Et eksempel på et datasett med to dimensjoner kan være en oversikt over antall dyr på en gård, se tabell 2.1. Skulle man for eksempel fremstille denne dataen som et søylediagram, ville diagrammet trenge to akser, x(dyr) og y(antall), for å kunne vise frem dataen.

Tabell 2.1 Data i 2D

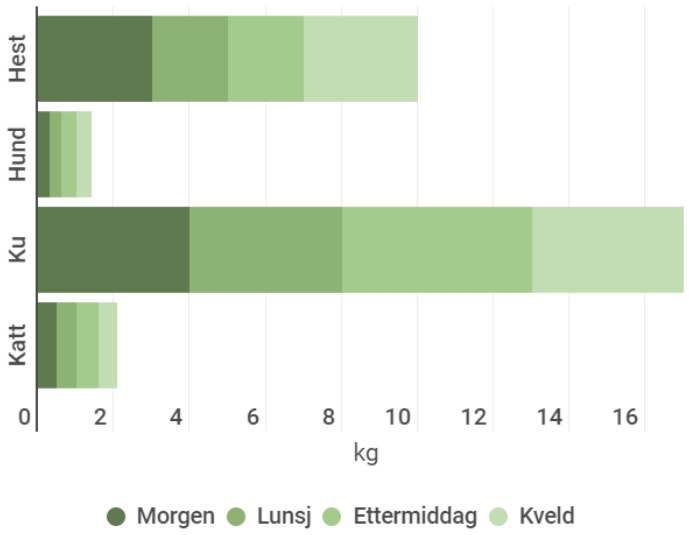
Dyr	Antall
Hest	3
Hund	2
Ku	4
Katt	3

Hvis vi derimot ser på hvor mye de forskjellige dyrene spiser, får vi data med tre dimensjoner, eksempelvis i tabell 2.2. Dette kommer av at hvert av tallene avhenger av både tidspunkt på dagen, hva slags dyr og hvor mye fôr dyret skal ha. Skulle man vist frem dette i et stolpediagram trenger man tre akser, én for å beskrive dyr, én for å beskrive når på dagen og én for å beskrive hva slags type fôr dyret får. Dette er illustrert med 3D-grafikk i figur 2.4.

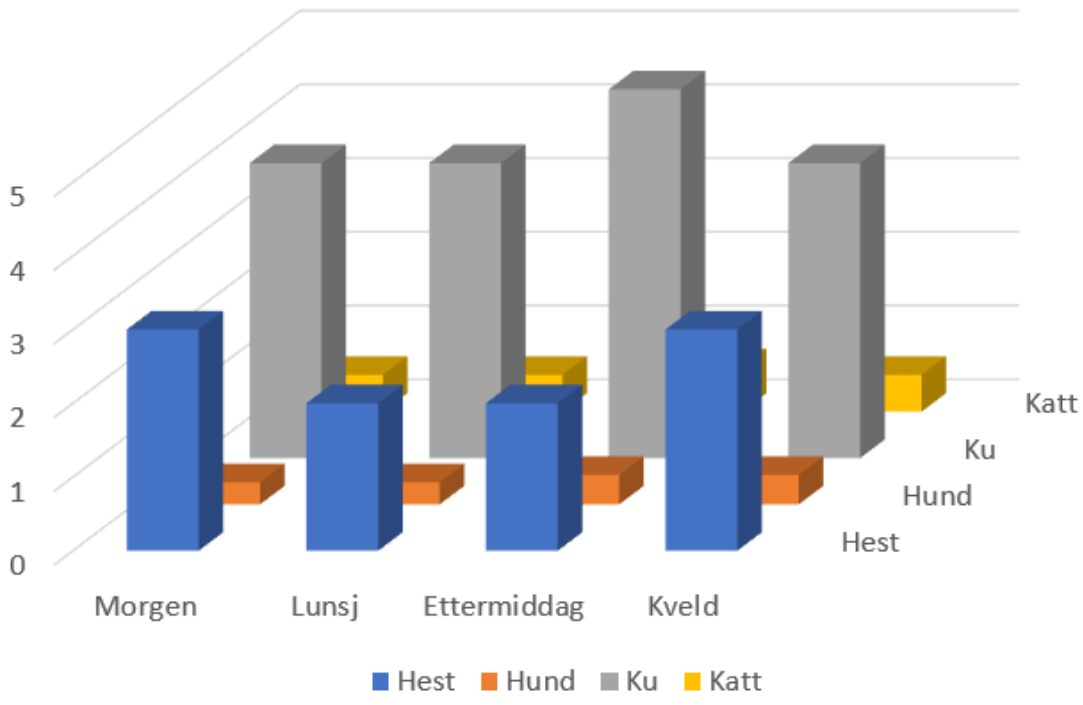
Tabell 2.2 Data i 3D

Dyr	Mengde	Tidspunkt
Hest	3	Morgen
Hest	2	Lunsj
Hest	2	Middag
Hest	3	Kveld
Hund	0,3	Morgen
Hund	0,3	Lunsj
Hund	0,4	Middag
Hund	0,4	Kveld
Ku	4	Morgen
Ku	4	Lunsj
Ku	5	Middag
Ku	4	Kveld
Katt	0,5	Morgen
Katt	0,5	Lunsj
Katt	0,6	Middag
Katt	0,5	Kveld

En annen måte å presentere disse dataene på, i 2D-grafikk, er ved å bruke et stablet bardiagram som illustrert i figur 2.3. Dette er et eksempel på en dimensjonal reduksjon. I et slikt diagram har man redusert antall akser til to, ved å representere mengden fôr et dyr skal ha som en blokk i på en bar.



Figur 2.8 *Stablet søylediagram i 2D-grafikk*



Figur 2.9 *Søylediagram i 3D-grafikk*

2.3 Webapplikasjoner

En webapplikasjon benytter en nettleser som klient, og kommuniserer med en tjener ved bruk av endepunkter hos både tjener og klient (*What is Web Application (Web Apps) and its Benefits*, u.d). Det særlig to aspekter ved utvikling av en webapplikasjon som tas stilling til; sikkerhet og universell utforming.

2.3.1 Sikkerhet

Kommunikasjon utført ved hjelp av internett-protokoller kan ha sikkerhetsmessige svakheter. I denne sammenheng har OWASP utviklet en oversikt over de ti mest vanlige sårbarhetene i webapplikasjoner, der man kan trekke ut at *injection*, eksponering av sensitiv informasjon, tredjepartskomponenter med ukjent sikkerhetsstatus, og dårlig konfigurasjon av sikkerhetsfunksjonalitet er noen av de mest vanlige (*OWASP Top Ten Web Application Security Risks, OWASP*, u.d).

En applikasjon som tillater opplastning av filer til en tjener, har særegne svakheter som det må tas hensyn til. Dersom applikasjonen tillater alle filtyper, vil det være mulig å angripe applikasjonen ved å laste opp og kjøre en eksekverbar fil med ondsinnet innhold, og slik utøve skade på maskinen som kjører tjeneren (*File uploads, Web Security Academy*, u.d). Et angrep kan også utføres ved å benytte filens navn til å skrive over filer som allerede eksisterer, eller ved å skaffe seg tilgang til maskinens mappestruktur (*File uploads, Web Security Academy*, u.d). Store filer kan også utøve skade på et system (*File uploads, Web Security Academy*, u.d).

2.3.2 Universell utforming

For å sikre at innhold på web-baserte ressurser har høy grad av tilgjengelighet, har W3C utarbeidet WCAG 2.1-standarden, som gir føringer for hvilke tiltak som kan gjøres for å tilgjengeliggjøre innhold (*Web Content Accessibility Guidelines (WCAG) 2.1*, u.d). Denne standarden stiller 35 krav som bør oppfylles.

Kognitiv overbelastning

Et annet virkemiddel for å tilgjengeliggjøre en applikasjons innhold er å sørge for at brukeren ikke rammes av kognitiv overbelastning. En slik overbelastning skjer når brukeren blir presentert en større mengde informasjon enn det hjernen klarer å prosessere (Byyny, 2016). På generell basis kan en bruker prosessere fra 5-9 forskjellige enheter med informasjon i arbeidsminnet til enhver tid (Byyny, 2016).

2.4 Smidig systemutvikling

Smidig systemutvikling er basert på de 12 prinsippene fra *the agile manifesto,* som vektlegger viktigheten ved samarbeid, hyppig leveranse av funksjonalitet, og god kommunikasjon med oppdragsgiver ('Agile Manifesto for Software Development, Agile Alliance', 2015).

En systemutviklingsprosess med en slik tilnærming vil levere små, kontinuerlige leveranser fremfor store lanseringer. Krav, planer og resultater evalueres kontinuerlig slik at teamene raskt kan reagere på endringer (What is Agile?, Atlassian, 2022). Det vektlegges særlig at planlegging og dokumentering ikke skal stjele for mye tid fra utviklingen av produktet ('What is Agile Software Development?', 2015). For å oppnå dette lar man en prosjektleder først bestemme prioriteringen til funksjonaliteten som skal implementeres, deretter organiserer utviklingsteamet selv detaljene rundt mindre oppgaver og oppdrag (What is Agile?, Atlassian, 2022). Dette fører til at teamet får mer eierskap til prosessen og produktet (What is Agile?, Atlassian, 2022).

2.4.1 Scrum

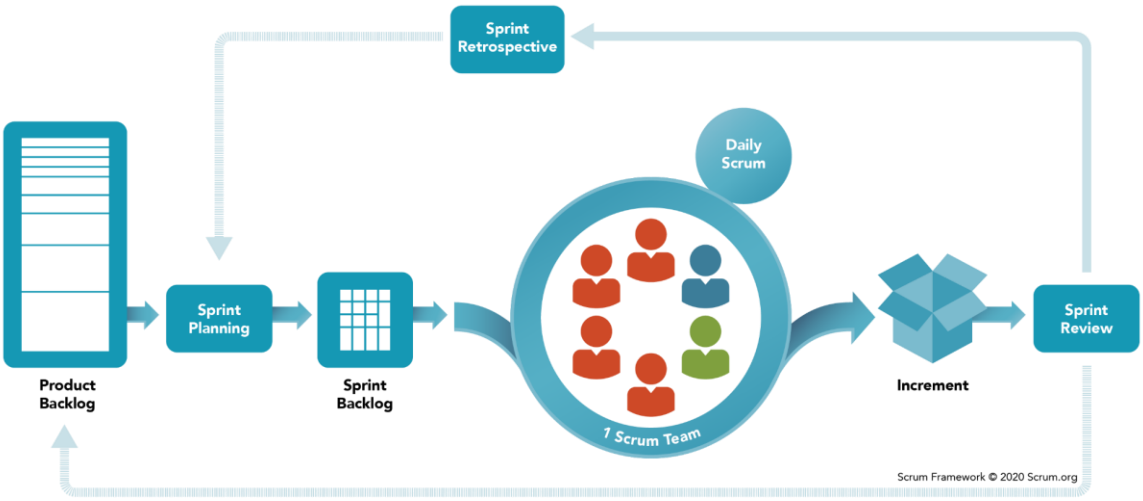
Scrum er et rammeverk utarbeidet med utgangspunkt i Manifestet for Smidig Programvareutvikling, og som tilbyr et utviklingsteam flere verktøy for å oppnå en god systemutviklingsprosess (*What is Scrum?*, u.d). Rammeverkets hovedtrekk er en iterativ utviklingsprosess, forhåndsdefinerte roller for alle involverte parter, samt artefakter og aktiviteter som skal bistå utviklingsprosessen (*Scrum Guide, Scrum Guides*, u.d).

Det er tre roller som blir utdelt under utviklingsprosessen; Scrum Master, produkteier, samt utviklingsteam. I rollen som Scrum Master har man ansvar for å legge til rette for en god utviklingsprosess, og at rammeverkets verktøy blir benyttet på en hensiktsmessig måte (*Scrum Guide, Scrum Guides*, u.d). En produkteier er som regel oppdragsgiveren, og har et særskilt ansvar for å definere krav og prioritering til produktet som skal leveres (*Scrum Guide, Scrum Guides*, u.d). Utviklingsteamet er involvert i utviklingen av produktet (*Scrum Guide, Scrum Guides*, u.d).

I tillegg til disse rollene tilbyr rammeverket også tre artefakter og forskjellige aktiviteter som støtteverktøy. De tre artefaktene er Produkt *backlog*, Sprintbacklog, samt et hovedmål for hver Sprint - som er rammeverkets definisjon for iterasjon. Aktivitetene utgjør de ulike delene av utviklingsprosessen, og består av følgende punkter (*Scrum Guide, Scrum Guides*, u.d);

- Sprint
- Sprint planleggingsmøte
- Daglig Scrum
- Sprintreview møte

- Sprint retrospektiv.



Figur 2.10 En Scrum-prosess visualisert.

Disse aktivitetene fordeler seg som illustrert i figur 2.4. Under Sprint planleggingsmøtet vil neste Sprint planlegges i samarbeid med produkteier, og resulterer i en Sprintbacklog som danner grunnlaget for arbeidet under kommende Sprint (*Scrum Guide, Scrum Guides*, u.d). Under en Sprint vil utviklerne gjennomføre daglige Scrummøter, der status for utført og planlagt arbeid legges frem (*Scrum Guide, Scrum Guides*, u.d). Ved slutten av en Sprint vil det leverte produktet presenteres i et Sprintreview møte, der tilbakemelding gis og veien videre blir diskutert (*Scrum Guide, Scrum Guides*, u.d). Før neste Sprint vil man så gjennomføre en Sprint retrospektiv der forrige Sprints styrker, svakheter og løsninger på disse blir lagt frem og diskutert (*Scrum Guide, Scrum Guides*, u.d).

2.5 Brukertesting

Brukertesting handler om å teste funksjonaliteten av produktet på brukere (*What is Usability Testing? (and What it Isn't), Hotjar*, 2022). Ofte baserer det seg på å observere brukeren mens de utfører oppgaver på produktet. Dette gjøres for å sikre at produktet løser de problemene det er tiltenkt, og at brukeren får den tiltenkte brukeropplevelsen (*What is Usability Testing? (and What it Isn't), Hotjar*, 2022). Målet med brukertesting er å avdekke aspekter av produktet som fører til forvirring, samt å finne muligheter for forbedring av produktet (*What is Usability Testing? (and What it Isn't), Hotjar*, 2022).

2.5.1 Kvalitativ brukertesting

Kvalitativ brukertesting handler om å samle inn kvalitative data gjennom brukertester. Metoden går ut på å bruke få testpersoner som gir dypere innsikt i problemstillingen, i motsetning til å bruke en større mengde testpersoner med mindre detaljerte data. Rundt brukertestene skal det være fleksible omgivelser og en fleksibel struktur, slik at testen kan justeres ved behov (Budiu, 2017). Kvalitative data består av observasjoner som

identifiserer om funksjonalitet er enkel eller vanskelig å bruke (Budiu, 2017). Kvalitative data gir en direkte evaluering av brukervennligheten til et system. Denne formen for brukertesting har som mål å sørge for at man kan ta informerte beslutninger for design og utforming, samt å identifisere problemer og finne løsninger (Budiu, 2017). Kvalitativ brukertesting vil gi resultater som baserer seg på observatørens observasjoner og personlige tolkninger.

2.5.2 Kvantitativ brukertesting

Kvantitativ brukertesting er en indirekte evaluering av brukervennligheten til et design (Budiu, 2017). Det baserer seg på prestasjoner på en gitt oppgave. Prestasjonene måles i tall, som for eksempel tid, suksess-rate eller antall feil. En av fordelene til kvantitative brukertester er at de har statistisk signifikans (Budiu, 2017). Derfor er kvantitativ brukertesting best når man har tilgang til mange testpersoner, der man kan samle inn store mengder data over tid, slik at de kan sammenliknes (Budiu, 2017).

2.6 Kvalitetssikring av kode

Under utviklingsprosjekter er det viktig å skrive kode av god kvalitet. Det er flere aspekter som inngår i å holde kvaliteten på koden god. I dette kapitlet beskrives bakenforliggende teori rundt de metodene som har blitt vektlagt gjennom prosjektet.

Testing av en kodebase utføres for å sikre at kodens funksjonalitet gir forventede resultater, uavhengig av brukerens handlinger. Man vil også sikre at funksjonalitet som blir implementert underveis ikke har ødeleggende virkning på eksisterende funksjonalitet (*What is Software Testing and How Does it Work?, IBM*, u.d). Et designmønster gir en generell, gjenbrukbar løsning for de vanlige problemene som kan oppstå i programvareutvikling ('Design Patterns, Set 1 (Introduction)', 2015). Mønsteret viser vanligvis relasjoner og interaksjoner mellom klasser eller objekter ('Design Patterns, Set 1 (Introduction)', 2015). Versjonskontroll brukes for å håndtere og organisere endringer i en kodebase (Ruparelia, 2010)

2.6.1 Testing

Det finnes mange forskjellige metoder og strategier for testing av kode, der disse blir valgt ut fra behovet til hvert enkelt system (*What is Software Testing and How Does it Work?, IBM*, u.d). En sentral fordel ved å ha høy dekningsgrad av tester er at feil kan avdekkes og utbedres raskt (*What is Software Testing and How Does it Work?, IBM*, u.d). Høy dekningsgrad er ofte assosiert med høy kodekvalitet (Best practices for writing unit tests - .NET, 2022).

Ved å kombinere enhetstesting med integrasjonstesting vil man ha et godt testgrunnlag for kodebasen. Enhetstesting utføres på mindre enheter, gjerne små deler eller metoder i koden. De har som hovedmål å isolere hver enhet i koden for å identifisere, analysere og fikse eventuelle defekter (Unit Testing, 2022). Integrasjonstester er større tester som sjekker kjernefunksjonalitet til applikasjonen fra ende til ende. I praksis betyr det å samle individuelle enheter og teste dem som en gruppe (Integration Testing, 2022). Hovedmålet til integrasjonstestene er å avdekke feil i interaksjonen mellom integrerte enheter (Integration Testing, 2022).

2.6.2 Avhengighetsinjeksjon

Avhengighetsinjeksjon (DI) er et programvaredesignmønster hvor avhengigheter injiseres inn i de aktuelle objektene som er avhengig av dem (Larkin, Smith and Dahler, 2022). Denne teknikken er en av metodene for å implementere prinsippet om invertering av kontroll (IoC) hvor rammeverket kontrollerer applikasjonens flyt samt opprettelsen av instanser (Martinez, 2022).

Både DI og IoC lener seg på prinsippet for avhengighetsinversjon som er en del av SOLID prinsippene innen objekt orientert programmering (Swistak, 2021) (Martinez, 2022). Prinsippet går ut på at moduler på høyt nivå ikke bør være avhengig av moduler på lavt nivå (Swistak, 2021). Begge skal avhenge av abstraksjoner (Swistak, 2021).

Selve implementeringen av avhengighetsinjeksjon skjer generelt på tre forskjellige måter; konstruktør injeksjon, grensesnitt injeksjon og setter injeksjon (Martinez, 2022). Uavhengig av hvilken metode som benyttes vil mønsteret frigjøre klassene fra sitt tidligere ansvar for sine avhengigheter (Martinez, 2022). Dette gjør klassene *decoupled* fra sine avhengigheter, som legger til rette for gjenbruk samt gjør koden lettere testbar (Martinez, 2022).

2.6.3 Versjonskontroll

Versjonskontroll er nødvendig ved alle større utviklingsprosjekter der flere utviklere jobber mot samme kodebase. Versjonskontroll brukes også for å lagre historikk, slik at tidligere versjoner av kodebasen lagres. Denne historikken gir mulighet for å rulle tilbake til en tidligere versjon av koden ved behov (Ruparelia, 2010). Den gir også oversikt hvor endringer ble introdusert, som kan bidra til å finne årsaken til problemer (Ruparelia, 2010).

Atlassian, u.d, beskriver flere positive aspekter ved bruk av VCS. Ved bruk av versjonskontroll vil man benytte et VCS til å lagre koden i en kodebase. Denne kodebasen inneholder all relevant informasjon om prosjektstrukturen. Ved arbeid med et slikt system brukes *branches* til å segmentere alt arbeid med funksjonalitet. Ved fullført arbeid vil disse *branchene merges* inn i kodebasens *master branch*, der det som regel vil finnes retningslinjer som må følges for at den tilførte koden skal bli akseptert til kodebasen. Dette gjør at det er mulig for flere utviklere å jobbe parallelt med samme kodebase.

Branch-strategi

Som en del av kvalitetssikring av kodebasen er det vanlig å benytte en strategi for håndtering av *branches*. Trunk-based utvikling er en slik strategi, som går ut på at man har en *master branch* som til ethvert tidspunkt skal være klar for å kjøres i produksjon (Haddad, 2022). Det vil så lages en ny *branch* for hver funksjonalitet som skal implementeres, og ny funksjonalitet blir hyppig integrert (Haddad, 2022). Ved å sette krav om at hver ny implementasjon sjekkes opp mot forhåndsdefinerte kvalitetskrav av et eller flere teammedlemmer, vil denne praksisen fungere som et ledd i kvalitetssikringen.

Målet med denne strategien er å eliminere *branches* som strekker seg over lang tid (Haddad, 2022). En trunk-based strategi er passende til mindre team som samarbeider tett gjennom utviklingsprosessen (Haddad, 2022).

Kapittel 3 Metode

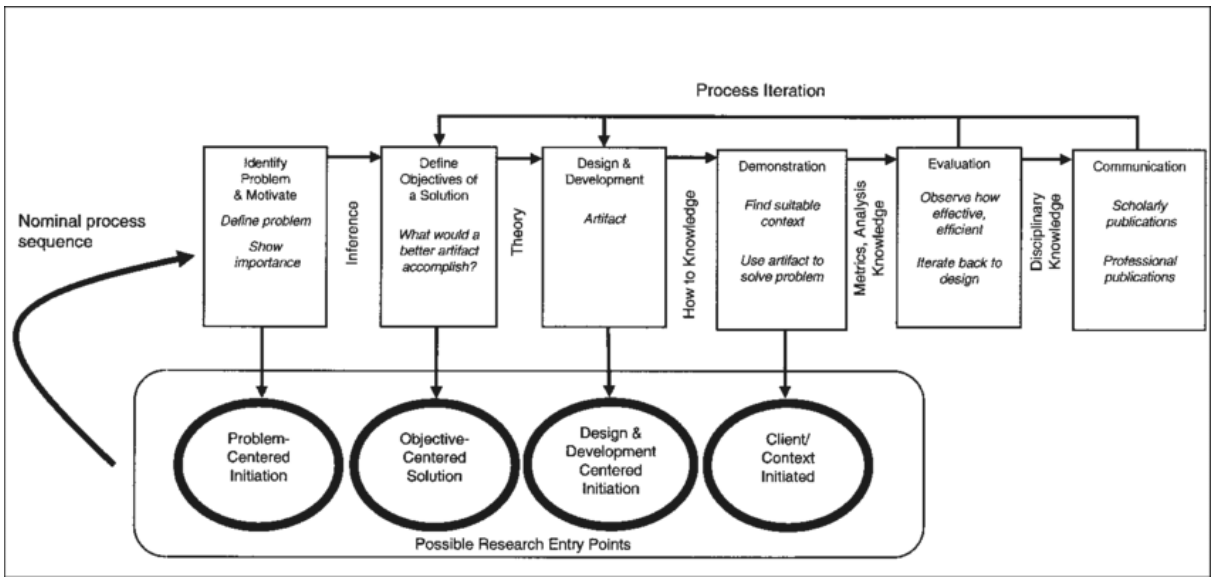
I forrige kapittel er relevant teori beskrevet nærmere. Hvordan metoder, teknologier og strategier benyttes beskrives nærmere i dette kapittelet. Det blir videre begrunnet kort om hvorfor disse valgene er tatt.

3.1 Forskningsmetode

Ved gjennomføring av en akademisk bacheloroppgave innen informasjonsteknologi kombineres utvikling med forskning. For å gjøre dette bruker gruppen forskningsmetoden Design Science Research hvor brukertester og litteraturstudie brukes for å samle kunnskap om løsningen utviklet oppnår det ønskede målet. Brukertestene gir innsikt i miljøet produktet er tiltenkt. Litteraturstudie gir innsikt i etablert kunnskap på området.

3.1.1 Design Science Research (DSR)

DSR har seks stadier; problemidentifikasjon og motivasjon, definere mål for en løsning, design og utvikling, demonstrasjon, evaluering, samt kommunikasjon som sees på figur 3.1. Metoden har som mål å generere kunnskap om hvordan ting kan og bør utformes for å oppnå et ønsket sett med mål. Denne kunnskapen er referert til som designkunnskap (DK) *(Brocke, J. & Hevner, A. & Maedche, A., 2020).*



Figur 3.1 Prosessmodell for DSR. (Brocke, J. & Hevner, A. & Maedche, A., 2020)

Første og andre del av denne metoden ble gjennomført av Tietoevry. De gjorde en problemidentifikasjon og fant motivasjonen for oppgaven. Behovet for å visualisere kompetanse ble identifisert. Videre definerte Tietoevry mål for løsningen (3. stadium). To resultatmål og tre effektmål ble utformet, se vedlegg 4 Powerpoint fra oppstartsmøte.

Gruppen gjennomfører tredje og fjerde stadium iterativt. I tredje stadium designer og utvikler gruppen en foreslått løsning, med tilbakemeldinger fra klienten og aktuelle brukere gjennom brukertester i fjerde stadium. Mot slutten av prosjektperioden blir resultatene evaluert helhetlig og kommunisert til aktuelle parter.

3.1.2 Litteraturstudie

Litteraturstudie blir benyttet for å faglig underbygge arbeidet med prosjektet. Det vurdert at å hente inn informasjon fra eksterne kilder ville være nyttig for prosjektet og prosjektfremdriften. Informasjon hentet fra eksterne kilder skal bekreftes fra flere hold, og det er viktig å være kritisk til hvor man henter inn informasjon fra. God kildeføring er også en viktig forutsetning for å kunne bruke litteraturstudie som en del av forskningsmetoden.

3.1.3 Kvalitative intervju og brukertesting

Informasjonsinnhenting

Kvalitative intervju med personer i målgruppen ble benyttet for å forstå utfordringen bedre, og dermed også hvordan man kan løse den på en god måte. En beskrivelse av gjennomføringen av brukertestene befinner seg i vedlegg 5 Oversikt over brukertester.

De kvalitative intervjuene ble utført i kombinasjon med kvalitative brukertester. En svakhet ved kvalitative intervju som forskningsmetode er at de produserer resultater som vil være basert på observatørens inntrykk og tolkninger (Raluca, 2017). Dette gir ikke utelukkende objektive data, i motsetning til de statistiske resultatene en kvantitativ brukertest ville produsert (Raluca, 2017).

De kvalitative brukertestene ble supplert med noe innhenting av kvantitative data, i form av en spørreundersøkelse utført i forbindelse med kvalitative intervju (se vedlegg 5). Det var et behov for å hente inn mer spesifikk informasjon fra det faktiske miljøet enn det man kunne finne ved hjelp av andres forskning. Det ble vurdert som nyttig for gruppen å samle inn tilbakemeldinger fra potensielle brukere av produktet. Tilgangen på testpersoner var noe begrenset.

Kvalitative brukertester gir svar på hvor problemene ligger (Raluca, 2017). Oppgavens omfang begrenset muligheten til å gjennomføre mange brukertester. Det ble dermed vurdert slik at kvalitative brukertester, supplert med innsamling av noe kvantitativ data, var bedre egnet for dette prosjektet.

Gjennomføring av brukertester

Oppgavens primære målgruppe er prosjektledere som arbeider med å sette sammen team, og dermed består storparten av testobjektene av personer som oppfyller dette kriteriet. Det brukertestes også på testpersoner med annen bakgrunn for å få flere perspektiver. I sammenheng med testen måtte også testpersonen svare på en rekke spørsmål.

Brukertestene ble gjennomført på lo-fi prototyper for applikasjonen. Dette ble gjort for å få tilbakemeldinger på brukervennlighet og funksjonalitet. Ved å fjerne alle forstyrrende elementer, som farger og annet design, unngår man at testobjektet fokuserer på dette.

Brukertestene ble gjennomført iterativt, i tre iterasjoner. Dette er nøyere beskrevet i vedlegg 5 *Oversikt over brukertester.* Grunnen til dette var at etter hvert som produktet og prosessen rundt utviklingen endret seg, endret også behovet for data seg. Det ble totalt gjennomført 10 brukertester og 11 innsamlinger av preferanser rundt

visualiseringer. I møte med oppdragsgiver 5. mai 2022 ble det gjennomført en gjennomgang av diagrammer for å samle inn mer data til diagrammene spesifikt.

3.2 Valg av teknologi

3.2.1 Webapplikasjon

Systemet ble utformet som en webapplikasjon. Ved å utforme produktet som en webapplikasjon sørger man for at produktet til enhver tid er tilgjengelig for kunden, og man kan oppdatere den i realtid. I en stor organisasjon som Tietoevry vil det være en fordel at tjenester brukt innad i organisasjonen er lett tilgjengelige for de ansatte som skal bruke den i arbeidet sitt. Oppdateringer og forbedringer kan også rulles ut kontinuerlig uten å direkte kreve handling, som oppdatering, fra brukeren.

En ulempe ved dette valget er at en konvensjonell webapplikasjon er avhengig av internettilgang for å fungere. Dette behovet for internett-tilkobling gjør også applikasjonen sårbar for angrep som beskrevet i kapittel 2.3.1 Sikkerhet.

3.2.2 Vue

Vue er et rammeverk basert på JavaScript, som brukes for å lage brukergrensesnitt på klientsiden (Kyriakidis, Maniatis and You, 2017). Gruppen vurderte at man kan få til all nødvendig funksjonalitet med Vue, uten store heftelser. I forhold til mer etablerte rammeverk som React, har ikke Vue tilgang til like mange tredjeparts-bibliotek og andre ressurser. Etter nærmere undersøkelser kom gruppen frem til at prosjektet ikke krevde mer funksjonalitet eller ressurser enn Vue kunne tilby, og dermed ble klienten bygget ved hjelp av dette rammeverket.

3.2.3 D3.js

For fremstilling av grafer i frontend falt valget på biblioteket D3.js. D3.js er et bibliotek basert på JavaScript, som legger til rette for å visualisere data gjennom ulike typer grafiske fremstillinger (Bostock, u.d.). Valget av bibliotek for grafisk fremstilling av data falt på D3.js av to ulike grunner. D3.js gir mange muligheter til å benytte seg av forskjellige typer grafiske fremstillinger, noe som man anså som viktig for prosjektet, da deler av prosjektet vil bestå av å teste ut forskjellige typer grafer. Dessuten tilbyr D3.js støtte for å lage grafene i vektorgrafikk, noe som var ønskelig fra produkteiers side, og som også er nevnt i vedlegg 2, Visjonsdokument.

3.2.4 ASP.NET

Tjeneren er satt opp som et ASP.NET Core-prosjekt, fungerer som REST API for klienten bygget i Vue. ASP.NET er et åpent kildekode-rammeverk, laget av Microsoft, for å bygge moderne nettapper og tjenester med .NET (What is ASP.NET?, .NET, 2022). Det var flere årsaker som gjorde at valget falt på ASP.NET. Det estimeres at omtrent en tredjedel av alle systemutviklere benytter seg av språket C# i verden i dag, og er dermed et høyst aktuelt språk å lære seg (Veeraraghavan, 2022). Videre brukes dette rammeverket av Tietoevry og dermed ga dette valget gruppen muligheten for teknisk støtte under prosjektperioden. Til slutt ble også bedriftens muligheter til å videreutvikle produktet etter endt prosjektperiode tatt i betraktning. Dette vil lettere la seg gjøre dersom det blir brukt teknologi som er kompatibelt med de systemene de har fra før.

3.2.5 Itext 7

En del av oppgavens omfang er å innhente informasjon fra PDF-filer. PDF er et filformat som lagrer informasjon om egenskapene til innholdet i filen, blant annet egenskapene til tekst som definerer ulike seksjoner i filen. Itext7 Core er et bibliotek som leverer funksjonalitet som gjør det mulig å hente ut og håndtere denne informasjonen (iText 7 Core, 2018), og var dermed nødvendig i arbeidet med å skape datagrunnlaget for visualiseringene.

3.2.6 Figma

For å raskt kunne utvikle og teste brukergrensesnittet, ble Figma benyttet som verktøy. Figma er et sky-basert designverktøy, som muliggjør samarbeid på tvers av teammedlemmer der løsningen blir oppdatert i sanntid (*The Power of Figma as a Design Tool*, u.d). En stor fordel er at Figma kan kjøres rett i nettleseren, som fører til bred støtte for ulike plattformer. Prototypene som utvikles kan også svært raskt deles med testbrukere, og tjenesten leverer en god løsning for testing av prototyper der testbrukeren kan trykke seg gjennom en interaktiv versjon av løsningen. Selve designverktøyet er svært brukervennlig, og det er lett å raskt sette seg inn i løsningen (*The Power of Figma as a Design Tool*, u.d).

3.2.7 Svg2Pdf.js

Svg2pdf.js er et bibliotek for JavaScript som konverterer SVG til PDF, og lar deg laste ned den genererte PDF-en. Dette er et bibliotek med åpen kildekode. Enkel implementasjon, samt effektiv nedlastning var deler av grunnen til at valget falt på dette biblioteket. Fordelen med at biblioteket har åpen kildekode er at man som utvikler kan inspisere denne koden, og dermed kan man være helt sikker på at informasjon ikke kommer på avveie. Skulle man ønske å øke sikkerheten enda mer ved en senere anledning, kan man også implementere samme funksjonalitet ved å hente koden og implementere selve kildekoden til biblioteket direkte. Dette gir full kontroll over både koden og eventuelle oppdateringer av koden.

3.3 Valg av utviklingsmetode

3.3.1 Scrum

Rammeverket Scrum ble valgt som smidig utviklingsmetode, først og fremst fordi gruppen ønsket å ha en iterativ og smidig prosess for å enkelt kunne oppdage og reagere på utfordringer og problemer. Videre tilbyr Scrum et godt dokumentert rammeverk, der artefaktene, aktivitetene og rollene er lette å forholde seg til. Gruppen så det som positivt å ha tydelig rolle- og ansvarsfordeling under utviklingen.

3.3.2 Azure Devops

Til deling av kildekode og strukturering av samarbeidet rundt produktutviklingsprosessen ble tjenestene Azure repos og Azure boards i Azure Devops benyttet. Azure Devops er

kompatibelt med Visual Studio og hosting hos Azure, noe som er fordelaktig for oppdragsgiver for videre utvikling og skalerbarhet.

Versjonskontroll

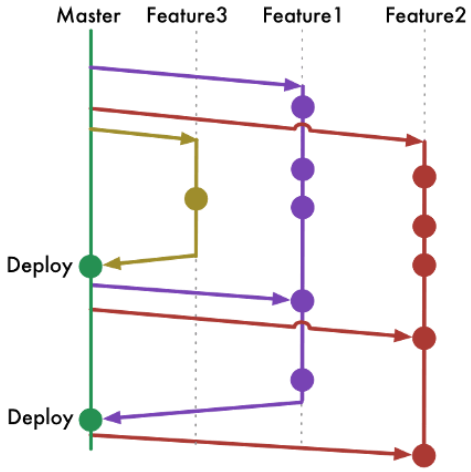
Tietoevry benytter seg av Azure DevOps til versjonskontroll, som gjør at det var et naturlig valg for denne oppgaven. Azure DevOps ga produkteier muligheten til å følge med på produktutviklingsprosessen, samt følge med på utviklingen av selve produktet. Ut over dette hadde gruppen definerte retningslinjer som skulle følges ved arbeid med kodebasen. Disse retningslinjene ble satt for å sikre kjørbar kode, høy kodekvalitet, og ikke minst en ryddig oversikt over det arbeidet som ble gjort til enhver tid.

Kvalitetssikring

Retningslinjene anga at ved fullført arbeid med en gitt funksjonalitet, feil eller test, skulle ny kode kjøres uten feilmeldinger, samt gås grundig gjennom av hvert gruppemedlem før den ble implementert i kodebasen.

Branch-strategi

Branch-strategien for prosjektet tar utgangspunkt i en *trunk-based* strategi. Dette ble vurdert som riktig valg for prosjektet da prosjektgruppen kjennetegnes av liten størrelse og tett samarbeid. *Master-branch* skulle heller ikke i produksjon under prosjektperioden, dermed ville ikke eventuelle feil ha store konsekvenser. Gruppen vurderte videre at det ville være mer effektivt at mengden *branches* ble holdt på et lavt nivå, samt at endringer ble implementert rett i *master branch*.



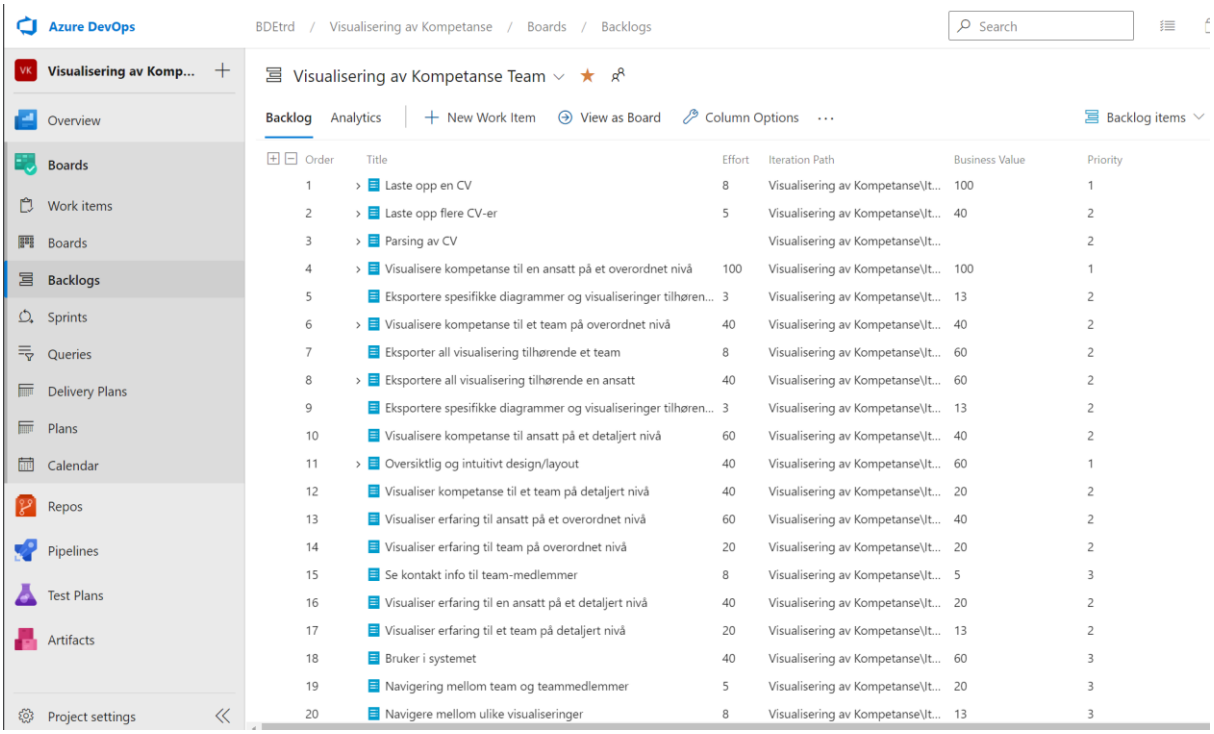
Figur 3.2 Trunk based utviklingsstrategi. (dev.to)

Prosjektstyring

Gjennom tjenesten Azure *boards* ble Sprints benyttet til å strukturere arbeidet iterativt. Mer spesifikt ble det benyttet produkt *backlog*, Sprintbacklog, Sprint tavle og *burndown charts*. Produkt *backloggen* ble satt opp med arbeidsoppgaver som fikk estimert tid og verdi. På figur 3.2 sees *Backloggen* i prioritert rekkefølge. Arbeidsoppgavene ble deretter tatt med videre i Sprintbackloggene, hvor de ble delt opp i mindre mer konkrete oppgaver. Nye oppgaver eller feil ble lagt til fortløpende under prosessen ved behov.

På Sprint tavlen ble det holdt oversikt over oppgavene sortert i kolonnene *To do*, *In progress* og *Done*. Automatisk genererte *burndown charts* ble forsøkt benyttet ut i prosessen ved å gi oppgavene en verdi for gjenværende tid. Når en oppgave ble satt til done ble *burndown charten* oppdatert.

Under utviklingen ble nye *branches* navngitt med ID-en til sin tilknyttede arbeidsoppgave i Sprint tavlen. Det førte til at det var lett å identifisere hvilke *branches* som tilhørte de ulike funksjonalitetene som skulle implementeres. Dette la til rette for at gruppen kunne arbeide dynamisk mellom forskjellige *branches*, som igjen gjorde det enkelt å bistå hverandre i utviklingen.



Figur 3.3 Produkt backlog i Azure DevOps Boards med estimert tid og verdi.

3.3.3 Databehandling

For å visualisere innholdet både i en enkelt CV og flere CV-er samtidig, trengs det strukturert data. CV-ene i seg selv er en måte å strukturere og visualisere kompetanse til de ansatte. Ved å plukke dem fra hverandre for å deretter sette dem sammen igjen i nye visualiseringer kan tolkeren raskt hente ut konkret informasjon. Kategorisering og sortering av dataene vil være verktøyet man bruker til å plukke CV-ene fra hverandre. Det er også viktig at dataene i kategoriene henger sammen, gjerne i kjente mønster for den som skal tolke dataene.

Kategorisering av data

For å kategorisere dataene på en måte som ivaretok og vektla flest mulig hensyn, valgte gruppen å vektlegge to hensyn i arbeidet. Det ene hensynet er hva som er mulig å få til med tanke på dataene som eksisterer i CV-en. Det andre hensynet er hva som i en ideell situasjon ville vært det absolutt beste produktet for produkteier, uten å se begrensninger som tilgang på data.

I forhold til datagrunnlaget i CV-ene, tas det utgangspunkt i strukturen til de standardiserte CV-ene. Det ble tatt utgangspunkt i prosjekter når man skulle hente data fra CV-ene. Informasjonen knyttet til et prosjekt omfatter kundenavn, varighet, rolle, beskrivelse, samt kompetanseområder. Disse kompetanseområdene blir deretter oppsummert i slutten av CV-en i kategoriene metoder, teknologi og verktøy. Gruppen definerte kategoriene:

- *Roller* til en ansatt vil være stillingstittel samt rollen man har hatt i ulike prosjekter. Eksempel på en rolle er utvikler eller UX designer.

- *Metoder* er spesifikke konsepter, ideer, fremgangsmåter eller oppskrifter man følger. Dette gjelder både for prosess og utvikling. Eksempler kan være Scrum, Kanban og universell utforming.
- Når det gjelder *teknologi*, er det tiltenkt digital teknologi. Dette omfatter grensesnitt, ulike programmeringsspråk, men også hardware.
- *Verktøy* sees her på som et håndfast system eller hjelpemiddel man bruker til å bygge en løsning eller bidrar på andre måter under utvikling. Det kan gjelde i prosess sammenheng gjennom f.eks. Azure DevOps, eller bruk av IDE som Intellij.

Når fokuset på hvilken data som trengs til ønskede visualiseringer, tas det utgangspunkt i hva en prosjektleder ønsker å se samt hvilke visualiseringer de ser på som intuitive og nyttige. Den ønskede visualiseringen vil være et resultat av kvalitative brukertester.

Prosessering av CV-er

For å omgjøre rådata fra en CV i PDF-format til datavisualisering i en webapplikasjon, ble det tatt utgangspunkt i de kategoriene som allerede var representert i CV-dataen. Dataene ble hentet ut ved hjelp av Itext 7 Core, som beskrevet i kapittel 3.2.5. Det ble så identifisert hvilken skrifttype og størrelse som definerte de ulike segmentene. Ved å kombinere denne kunnskapen med ordgjenkjennelse, kunne man skille ut informasjon knyttet til de ulike segmentene.

Datakvalitet

Innhold og kvalitet på dataen i CV-ene varierer. De oppdateres av de respektive ansatte i bedriften og benyttes ofte i overgangen mellom to prosjekter. Da brukes gjerne CV-ene til innsalg av kompetansen til den aktuelle kandidaten. Dataene er derfor i flere tilfeller ikke oppdatert, mangelfull og mangler relevant informasjon. Kompetanseområder er for eksempel ikke konsekvent inkludert i alle CV-er. Videre er kategoriseringen i flere tilfeller inkonsistent, hvor teknologier, metoder og verktøy glir over i hverandre. Samlet gir dette utfordringer med uthenting av data fra CV-ene.

3.4 Arbeids- og rollefordeling

Under utviklingen av produktet har gruppen fordelt roller som følgende:

Nora E. Jansrud - Møteansvarlig. Har i utviklingen hatt spesielt fokus på *frontend* og implementering av diagrammer.

Linda K. Larsen - Ansvarlig for kvalitetskontroll av kildekode. Har i utviklingen arbeidet *full-stack* med spesielt fokus på prosessering av CV-er, *backend* og design av *frontend*.

Jenny F. Blindheimsvik - Innleveringsansvarlig og *Scrum Master* innad i teamet. I utviklingen har Blindheimsvik hatt spesielt fokus *backend* og testing.

Roar Gjøvaag (Tietoevry) - Produkteier.

Utviklingsteamet består av Jansrud, Larsen og Blindheimsvik. Rollefordelingen ble fordelt på grunnlag av gruppens ønsker, styrker og kompetanseområder. De ulike rollene med

ansvarsområder finnes mer utdypende beskrevet i arbeidskontrakten, vedlegg 1 Prosjekthåndbok.

Gruppen valgte å tildele Blindheimsvik rollen som *Scrum Master*, til tross for at det er anbefalt å ha en utenforstående i denne rollen (Andersen, 2016). Grunnet manglende ressurser hadde ikke gruppen tilgang på en ekstern person, men så det likevel hensiktsmessig å ha en person ansvarlig for prosjektstyringen, samt å lede Scrum aktivitetene.

Kapittel 4 Resultater

4.1 Vitenskapelige resultater

For å svare på problemstillingen:

> «Hvordan kan en webapplikasjon for visualisering av kompetanse utvikles for å fasilitere sammensetning av et profesjonelt team?»

ble det gjennomført tre aktiviteter; kategorisering av data, kvalitative intervjuer og en datainnsamling på preferanser for diagrammer. Disse aktivitetene knyttes direkte opp mot effektmålene definert i kapittel 1.1 Relevans.

For å nå effektmål 1 ble det gjort en undersøkelse rundt preferanser på diagrammer, presentert i kapittel 4.1.2 Datavisualisering. Kategorisering av data ble gjennomført for å nå effektmål 1 og 3. For å oppnå effektmål 2 og 3 ble det gjennomført kvalitative intervju for å kartlegge og lære mer om prosessen med teamsammensetning, samt å avdekke områder med forbedringspotensialer.

I dette kapitlet vil resultatene fra arbeidet med å kategorisere data, samt resultatene fra innsamlingen av preferanser om datavisualisering presenteres. Deretter vil resultater fra kvalitative intervjuer.

4.1.1 Kategorisering av data

Kategorisering av kompetanseområder i CV-ene ble gjort for å skape et verktøy som kunne bidra til å objektivisere teamsammensetting, samt å tydeliggjøre hvilken kompetanse et individ eller team har (effektmål 1 og 3). Erfaringer fra brukertester, samt en undersøkelse av hvordan Tietoevry kategoriserer kompetanse, resulterte i at dataene ble delt opp i underkategorier. Dette ble også gjort fordi det kom frem fra brukertester at det var mer utfordrende å tolke diagrammer som har mange forskjellige kategorier i datasettet. Basert på Tietoevrys måte å kategorisere kompetanse, ble kategoriene *roller, metoder, teknologi og verktøy* trukket ut som kategorier for å kategorisere kompetanseområder til ansatte. Under brukertester fra runde 2 ble også referanser trukket ut som en kategori. Dette resulterte i at dataene som hentes ut fra CV-en kategoriseres i fem forskjellige kategorier; roller, metoder, teknologi, verktøy og referanser. Det ble ikke hentet ut utdanning fra CV-ene. Dette kom etter funn i brukertestene som indikerte at for de aller fleste er relevant arbeidserfaring viktigere enn utdanning.

Et annet funn fra bearbeiding av data, var at datasettene måtte kategoriseres ut fra hvor mange dimensjoner de hadde. Kategorisering av data etter dimensjoner på datasettet ble gjort for å kunne ta mer objektive beslutninger rundt teamsammensetning.

4.1.2 Datavisualisering

For å på best mulig måte kunne svare på problemstillingen, og lage et produkt som oppfyller kravene satt av oppdragsgiver, var det nødvendig å samle inn data rundt datavisualisering. Behovet for å samle inn data om preferanser og tolkninger av

visualiseringer er spesielt tett knyttet opp mot å gjøre det tydelig hvilken kompetanse et individ eller team innehar.

Dataene som presenteres i dette kapitlet er ikke basert på en stor kvalitativ undersøkelse, men en innsamling av data fra 11 personer. 10 av disse var i form av en standardisert brukertest. Den siste datainnsamlingen ble gjort i forbindelse med et møte med oppdragsgiver 05. mai 2022 (vedlegg 1, Prosjekthåndbok). Innsamlingen av data og metoden for dette er både beskrevet i kapittel 3.1.2 Kvalitative intervju og brukertesting, samt i vedlegg 5 Oversikt over brukertester.

Resultater som omhandler datavisualisering er hentet på to måter, gjennom litteraturstudier og brukertester. Litteraturstudiet hentet inn kvalitativ informasjon om datavisualisering, og supplerer med data som ikke er hentet gjennom brukertestingen.

Resultatene fra litteraturstudiet er hovedsakelig basert på to kilder; Edward R. Tuftes bok *The Visual Display of Quantitative Information* og et skriv av Jenny V. Freeman og Steven A. Julious som omhandler grafisk fremstilling av data. Ut fra dette trekkes konklusjonen at felles for de fleste definisjoner av hva en god grafisk fremstilling er at fremstillingene bør være enkle, og fremstille så mye informasjon som mulig uten bruk av unødvendige forstyrrende elementer. Samtidig er det viktig å tenke på hvordan fremstillingen oppfattes for den som skal lese dem.

Med brukertestene ble det hentet inn både kvalitativ og kvantitativ data. De kvalitative dataene ble brukt direkte i videre utvikling av produktet, samt til å underbygge noen av resultatene fra de kvantitative dataene. De kvantitative dataene ble også brukt som grunnlag for utforming av selve datavisualiseringen, spesielt med tanke på hvordan dataene skulle visualiseres. Resultatene fra disse brukertestene vil bli presentert under. Først vil resultatene fra datainnsamlingen for en person presenteres. Deretter vil resultatene for team presenteres.

Enkeltperson

For enkeltperson ble det brukt datasett med tre forskjellige størrelser; lite, middels og stort. Dette ble gjort med bakgrunn i en hypotese om at den fortrukne visualiseringen ville avhenge av størrelsen på datasettet. Det lille datasettet inneholdt roller, det middels store verktøy og det største datasettet omfattet teknologier. Under vises en oversikt over hvor mange ganger det tilhørende diagrammet ble trukket frem som en av de beste diagrammene under brukertestene. Dataene er kategorisert slik at dersom det har blitt trukket frem 3 ganger eller mindre i brukertestene, scorer det lavt. Har det blitt trukket frem fire til fem ganger scorer det middels. Har det blitt trukket frem seks eller flere ganger scorer det høyt.

Tabell 4.1 Populariteten til diagrammene ved kompetansen til én ansatt.

Datasett	Lite	Middels	Stort
Søylediagram	6	5	7
Radardiagram	4	2	1
Boblediagram	6	3	6
Treemap	2	5	5

Som vi ser av tabell 4.1, er søylediagrammet mest populært for alle datasett. Likevel er det relativt små forskjeller mellom de forskjellige diagrammene. Ved en oppsamling av

data som denne kan det se ut som søylediagram er den beste løsningen, men setter man dette i perspektiv med resultatet fra alle brukertestene, viser det seg at vurderingen av diagram baserer seg på hovedsakelig to ting; hvilke diagrammer du er vant til fra før og personlig preferanse. Radardiagrammet er et godt eksempel på personlig preferanse. Der testpersonen i brukertest 3 synes radardiagrammet er uoversiktlig og vanskelig å lese, mente testpersonen i brukertest 5 at det er en fin visualisering.

Team

For team ble det i likhet med enkeltperson brukt tre datasett av ulik størrelse. Tallene er hentet på samme måte som ved enkeltperson. Her varierte størrelsen både på antall data og antall personer som er i teamet. For team har skalaen for inndeling av data vært følgende; lav: 0-2, middels: 3-5, høy: 6<.

Tabell 4.2 Populariteten til diagrammene ved kompetansen til et team.

Datasett	Lite	Middels	Stort
Søylediagram	3	5	4
Stablet søylediagram	8	6	5
Radardiagram	2	0	1
Boblediagram	3	4	4
Treemap	0	4	3

Som illustrert i tabell 4.2 er det mye data som faller i kategorien middels. Den klare favoritten var det stablede søylediagrammet. En utfordring med brukertestingen på team var at det er vanskelig å visualisere store datasett på en god måte. Dette ble også kommentert av flere testpersoner under brukertestene.

Et annet hensyn å ta når man skal visualisere et team er hvorvidt man kun skal vise data for teamet som en total, eller om det skal være en indikasjon på hvordan erfaringen fordeler seg innad i teamet. For denne fremstillingen er det kun det stablede søylediagrammet som visualiserer hvordan erfaringen fordeler seg innad i teamet. Denne kom frem som en populær fremstilling, men på veldig store datasett syntes mange den var uoversiktlig. Dette førte til at den tok lengre tid å prosessere.

4.1.3 Fasilitering av prosess for teamsammensetting

Resultatene rundt fasilitering av prosessen for teamsammensetning er hentet fra brukertest 4 til brukertest 10 og baserer seg i stor grad på svarene samlet inn i intervjudelen før og etter brukertesten (Vedlegg 5). Resultatene i dette underkapitlet vil deles opp i to; en del som presenteres prosessen for teamsammensetning og hvordan dette gjøres, og en del som presenterer resultatene knyttet til hvordan produktet som har blitt utviklet kan bidra til å fasiltere denne prosessen. Disse resultatene sees i sammenheng med å effektivisere beslutningene rundt teamsammensetning, samt å gjøre dem mer objektive.

Sammensetting av team

Prosessen for teamsammensetning er en kompleks prosess som varierer fra bedrift til bedrift. Av de ti brukertestene som har blitt gjennomført, har seks av ti vært ansatte hos Tietoevry. En del av informasjonen som er innhentet gjennom intervjuene vil derfor være spesifikt tilknyttet denne bedriften. Det er i tillegg gjort intervju med personer i relevante stillinger i andre bedrifter, for å skaffe et bredere grunnlag og forståelse av prosessen for teamsammensetning.

Grunnlaget for de fleste prosesser med teamsammensetting er at det er behov for å løse en oppgave. Denne oppgaven krever spesifikk kompetanse, som det er viktig at medlemmene i teamet innehar. Det er ofte en person med roller som prosjektleder som har ansvaret for å sette sammen riktig folk til et team med riktig kompetanse og erfaring. I tillegg til kompetanse og erfaring, er også aspekter som kommunikasjonsevne og hvem som jobber godt med hvem viktige.

For å finne de riktige folkene til jobben, svarer et flertall av testpersonene at de baserer seg på hva de selv vet om de ansatte. Noen har også et CV-system som fungerer som en database for kunnskap hos enkelte bedrifter. I disse tilfellene svarer fortsatt en del at de baserer seg på hvem de kjenner, og supplerer ved å bruke CV-systemet. I noen tilfeller kjenner ikke prosjektlederen noen med tilstrekkelig kompetanse, og må lete opp den riktige personen. Slike tilfeller løses som oftest ved at man prater med andre man kjenner i bedriften, og forhører seg om de vet om noen med riktig kompetanse.

Av de 10 brukertestene som ble gjennomført, hadde fem av testpersonene arbeidsoppgaver som innebar teamsammensetting. I alle disse brukertestene ble spørsmålet «Synes du metoden for sammensetning av team i dag er god?» I alle fem av disse testene svarte testpersonen enten «nei» eller kom med forslag til forbedringer fremfor å svare ja. Flere nevner at prosessen med å sette sammen team er både tidkrevende og personavhengig.

Fasilitering av prosessen

Under brukertestene kom det frem at om man har en database med kompetanse hvor man kunne funnet aktuelle personer uten å basere seg på å spørre andre, ville dette slått positivt ut. Et annet aspekt som kommer frem er en oversikt over tilgjengelighet blant ansatte, slik at man vet hvilke ressurser som er tilgjengelig i hvilke tidsperioder og kan planlegge ut fra dette. Det trekkes også frem at visualiseringen gjør at man som prosjektleder enklere kan skaffe seg en oversikt. En annen fordel ved visualiseringene som nevnes er at man enklere kan oppdage kunnskapshull hos teamet som helhet, og dermed finne noen til å tette disse hullene.

Når testpersonene ble spurt hvordan de så for seg at et verktøy likt det de fikk prøve under brukertesten kunne bidra til å fasilitere prosessen med teamsammensetting, var et fellestrekk for mange svar at man ønsket seg flere utbedringer. Det som ble trukket frem oftest var tilgjengelighet blant de ansatte.

Under brukertestene var det flere fordeler ved produktet som ble trukket frem ved bruk av produktet i prosessen med teamsammensetting. At verktøyet kan gjøre det mulig å hente informasjon om kompetansen til ansatte uten å måtte benytte seg av ressurser i egne nettverk, samt en bedre oversikt over hvilke ressurser et team innehar og hvilke de trenger og muligheten for å bruke visualiseringene i kommunikasjon med produkteier og ved salg var aspekter som ble trukket frem gjentatte ganger. Et annet aspekt som nevnes av flere er tidsbesparelsene ved å kunne bruke mindre tid på å lete seg frem til de rette ansatte.

4.2 Ingeniørfaglige resultater

De ingeniørfaglige resultatene tar utgangspunkt i de funksjonelle og ikke-funksjonelle kravene satt i samarbeid med oppdragsgiver. Disse kravene er basert på effekt- og resultatmålene definert i Vedlegg 1 Prosjekthåndboken. I vedlegg 9 Kildekode, ligger systemet, og i Vedlegg 6 Systemdokumentasjon, ligger beskrivelse av installasjon og kjøring av prosjektet.

4.2.1 Funksjonelle krav

De funksjonelle kravene ble satt i samarbeid med produkteier i oppstartsmøte for bachelor 19. januar 2022 (vedlegg 1 Prosjekthåndbok). Med utgangspunkt i listen over funksjonelle krav i vedlegg 2 Visjonsdokument, ble det utarbeidet seks *user stories*. Denne listen med *user stories* gir et korrekt bilde av de funksjonelle kravene og deres prioritering.

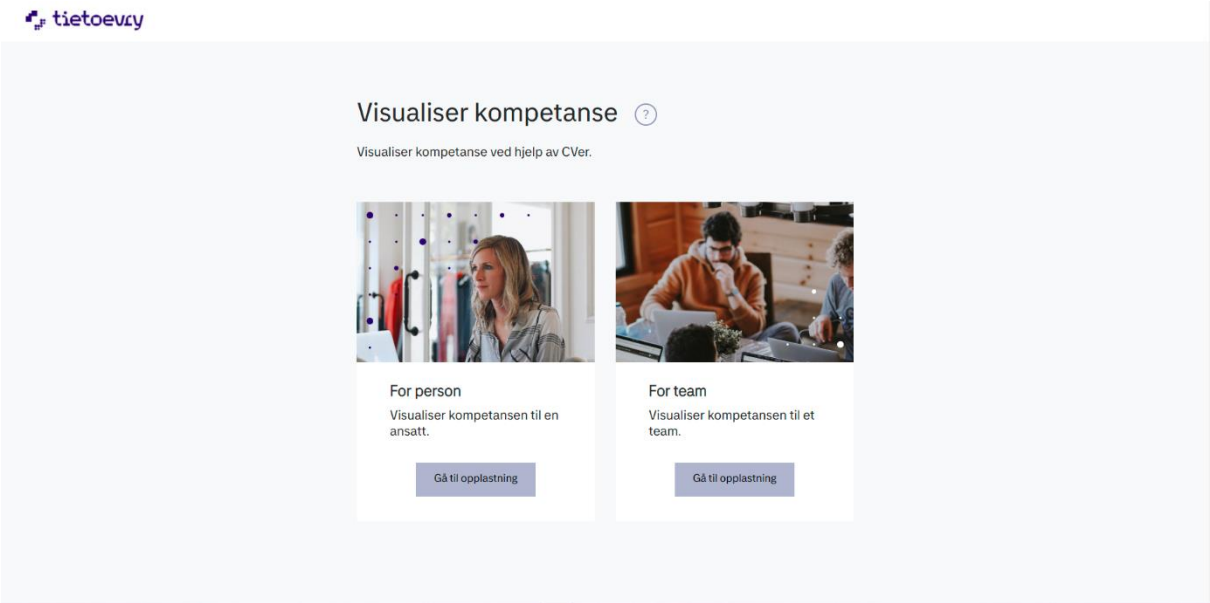
Som prosjektleder

Ønsker jeg...	slik at...	Prioritet	Akseptansekriterier	Status
å laste opp en CV	informasjonen den inneholder blir visualisert	1	Mulig å laste opp en cv	100%
			Mulig å lagre og behandle informasjonen i CV-en	100%
			Sum:	**100%**
å se en ansatt sin kompetanse og erfaring visualisert	jeg har oversikt over deres kompetanse og erfaring	2	Hull i kompetansen kan avdekkes	100%
			Kompetansen visualiseres i kategorier og underkategorier	50%
			Erfaringen innenfor ulike kategorier visualiseres	100%
			Sum:	**83%**
å se et team sin kompetanse og erfaring visualisert	jeg har oversikt over teamets kompetanse og kan lettere formidle dette	3	Flere CV-er kan lastes opp samtidig. (2-7 stk)	100%
			Evt. hull i kompetansen til teamet visualiseres	100%
			Kompetansen visualiseres i kategorier og underkategorier	50%
			Erfaringen til teamet innenfor de ulike kategoriene visualiseres.	100%
			Sum:	**88%**
å kunne laste ned visualiseringer	jeg enkelt kan lagre og bruke dem senere	4	Eksportering til vektrografikk	100%
			Mulig å laste ned all visualisering tilhørende ansatt og team	0%
			Mulig å laste ned spesifikke diagrammer og visualiseringer	100%
			Sum:	**67%**
å lagre informasjon om ansatte i systemet	at de lettere kan settes sammen i ulike team	5	Innlogget som prosjektleder kan man lagre, oppdatere og slette ansatte i systemet.	0%
			Oppretting og oppdatering av ansatte skjer gjennom parsing av CV-er	0%
			Sum:	**0%**
å kunne sette sammen team	jeg enkelt kan sette sammen nye team og evt. sammenligne ulike sammensetninger	6	mulig å sette sammen team fra ansatte lagret i systemet.	0%
			Sum:	**0%**
			Total måloppnåelse:	**56%**

Figur 4.1 User Stories

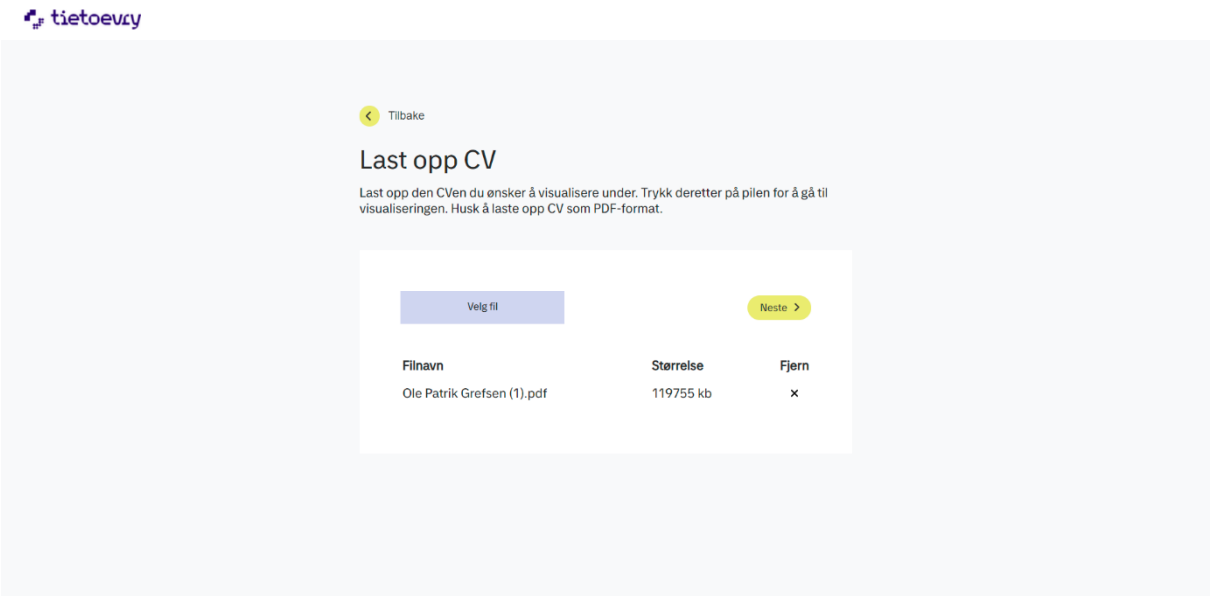
Figur 4.1 viser prioritert rekkefølge, omfang og grad av måloppnåelse for hver *user story*. Produktet ble utviklet som en webapplikasjon basert på Vue og ASP.NET. Ved oppgavens slutt var 56% av akseptansekriteriene oppnådd, der man kan se at *user-storiene* med høy prioritet også har høy grad av måloppnåelse. De to *user-storiene* med 0% måloppnåelse ble ikke prioritert på noe punkt under utviklingsprosessen, dermed er det ingen resultater å vise til.

Laste opp en CV



Figur 4.2 Home view

Applikasjonen starter på startsiden. Her kan brukeren velge om den ønsker å visualisere for én person, eller for et team. Deretter kommer man til opplastningssiden.



Figur 4.3 View for CV-opplastning

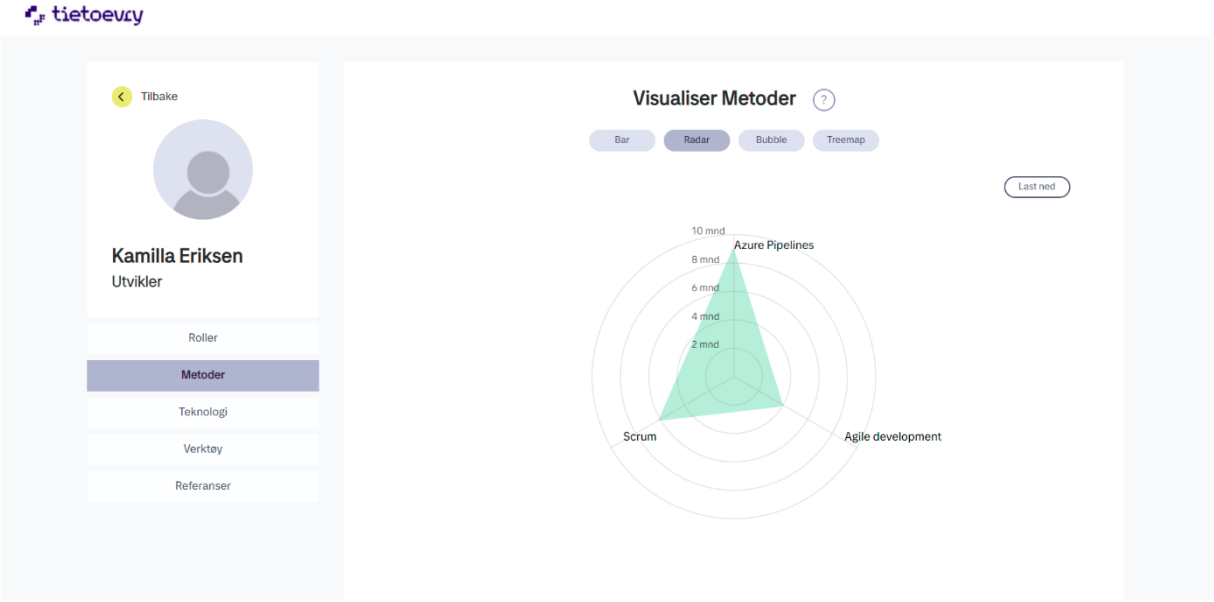
Som prosjektleder har man mulighet til å velge mellom å laste opp én CV for å visualisere en enkelt ansatt, eller flere CV-er for å visualisere et helt team. Denne CV-en må være utformet etter fastsatt mal, se vedlegg 8 Testdata, generert i CV-partner. Dersom prosjektlederen velger å visualisere én enkelt ansatt vil det kun være mulig å laste opp én CV om gangen. Her hindrer systemet brukeren fra å laste opp mer enn en fil, samt filer i andre format enn PDF. Ved opplastning av team har prosjektlederen mulighet til å velge flere PDF-filer for opplastning.

De valgte filene blir presentert i en liste der prosjektlederen kan se navn og størrelse på den opplastede filen, for å identifisere eventuelle tomme filer. Filnavnene inneholder

navnet på den ansatte representert i den opplastede CV-en, og prosjektlederen har dermed oversikt over hvem CV-en tilhører. I denne listen har prosjektlederen mulighet til å fjerne uønskede filer ved behov.

Systemet i backend returnerer ikke umiddelbart de prosesserte dataene fra de innsendte PDF-filene. For å kommunisere dette til prosjektlederen er knappen som fører videre til visualisering deaktivert helt til klienten mottar data fra tjener. Når denne dataen er mottatt vil knappen aktiveres, og prosjektleder kan trykke seg videre inn til visualisering av enten en enkelt ansatt eller et team.

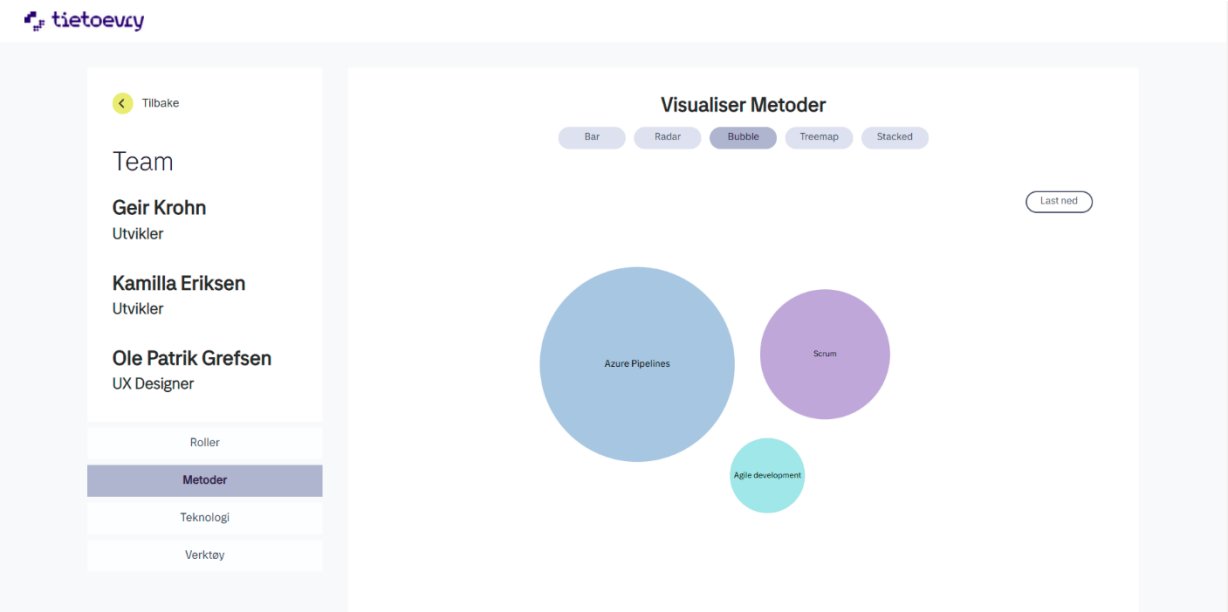
Visualisering av en ansatts kompetanse



Figur 4.4 View for visualisering av kompetansen til én ansatt.

På viewet som presenterer visualisering av en ansatts kompetanse kan prosjektlederen finne informasjonen som blir hentet ut fra CV-en. Til venstre kan prosjektlederen se navn og stillingstittel til den ansatte. Det er lagt inn et eksempelbilde, til videre implementasjon. Under denne oversikten finnes det en liste med valg av kompetansekategorier; roller, metoder, teknologi og verktøy. Prosjektlederen kan velge hvilken kompetanse som skal visualiseres. Ved trykk på disse knappene vil innholdet i visualiseringen endres basert på kategori, og kategorien er markert så lenge dette valget er aktivert. Prosjektlederen har også mulighet til å velge mellom fire forskjellige diagrammer for hver kategori. Nederst på siden vises referanser, med oversikt over tittel, oppdragsgiver og varighet for prosjektene vedkommende har jobbet på.

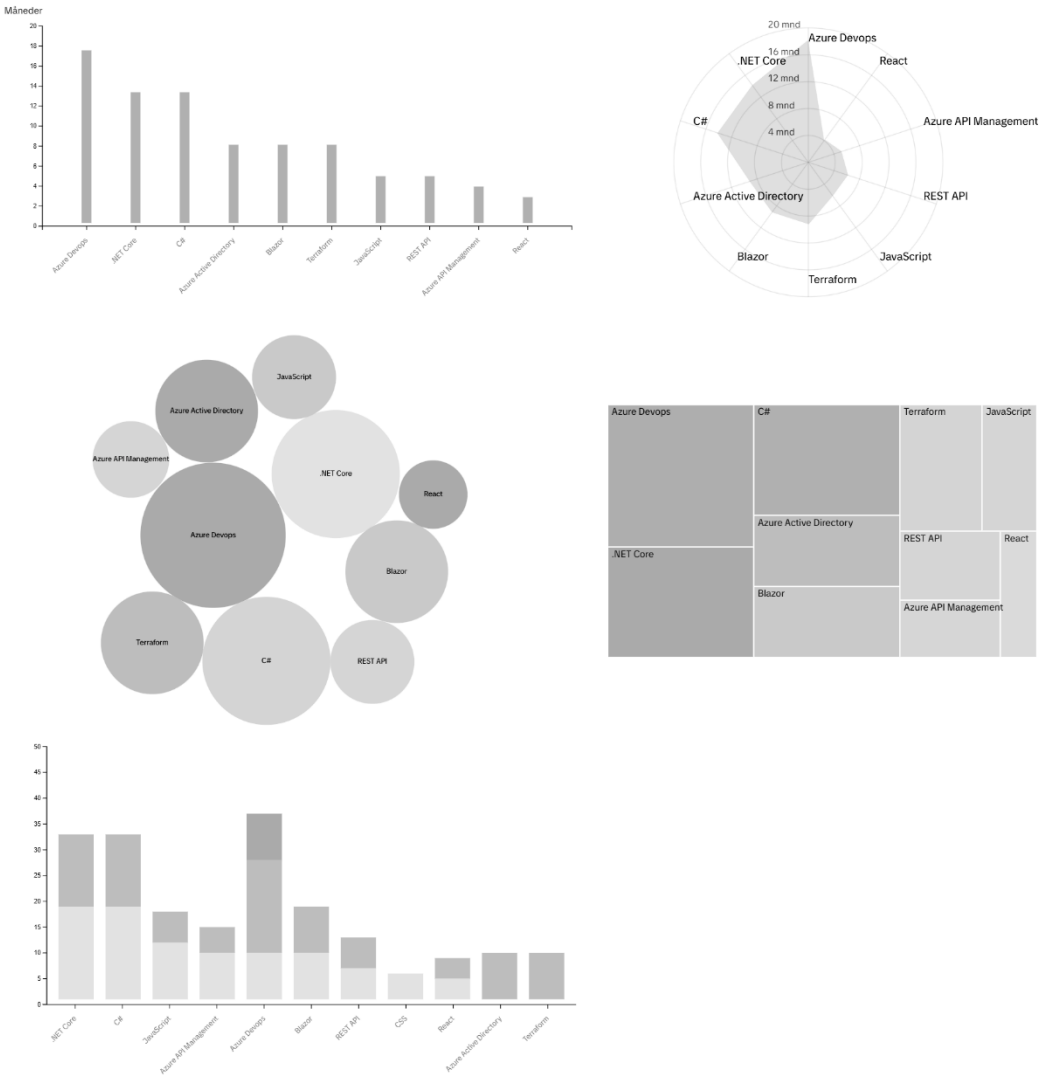
Visualisering av et teams kompetanse



Figur 4.5 View for visualisering av kompetansen til et team

Siden som presenterer visualisering av kompetansen innad et team har mange likhetstrekk med siden for ansatt, med noen tilpasninger for å tilby informasjon som er relevant for team. Til venstre kan prosjektlederen skaffe seg en kort oversikt over teammedlemmene. Prosjektlederen kan velge mellom fire forskjellige kompetanseområder; roller, metoder, teknologier og verktøy. Kompetanseområdene visualiseres med fem forskjellige diagram; søylediagram, stablet søylediagram, radardiagram, boblediagram og treemap.

Nedlastning av visualiseringer



Figur 4.6 Diagrammer til nedlastning

Prosjektlederen har muligheten til å laste ned valgt diagram som en PDF-fil med diagrammet som vektor-grafikk i SVG-format. Se tabell 4.3 for alle mulige kombinasjoner som kan lastes ned for enkeltansatte og team.

Tabell 4.3 Oversikt over nedlastbare diagrammer

Enkeltansatt: P Team: T		Søylediagram	Stablet søylediagram	Radardiagram	Boblediagram	Treemap
	Rolle	P, T	T	P, T	P, T	P, T
	Metoder	P, T	T	P, T	P, T	P, T
	Teknologi	P, T	T	P, T	P, T	P, T
	Verktøy	P, T	T	P, T	P, T	P, T
	Referanser	-	-	-	-	-

4.2.2 Ikke-funksjonelle krav

De ikke-funksjonelle kravene er fastsatt av vedlegg 2 Visjonsdokument. Disse kravene er basert på FURPS, der produktet oppfyller disse kravene i varierende grad. Disse kravene ble prioritert i samarbeid med produkteier i møte 28. januar 2022 (vedlegg 1 Prosjekthåndbok).

Functionality

Produktet ble utviklet som en webapplikasjon, i henhold til kravene satt i vedlegg 2 Visjonsdokument. Applikasjonen prosesserer og fremstiller informasjon basert på data fra innsendt CV. Produktet benytter *localstorage* til lagring av data, dermed var det ikke behov for å oppfylle kravene i forhold til datalagring i database. Produktet har ingen brukerhåndtering, slik at også sikkerhetskravene i forhold til dette, samt GDPR, faller bort. Produktet benytter HTTPS-protokollen for kommunikasjon mellom tjener og klient, som spesifisert i visjonsdokumentet.

Sikkerhetstiltak utover dette står beskrevet i vedlegg 6 Systemdokumentasjon.

UX - User experience

Produktet tilfredsstiller en del av de aktuelle kravene gitt av WCAG 2.1-standarden. En oversikt over hvilke krav som er tilfredsstilt kan sees i vedlegg 7 WCAG Sjekkliste.

Produktets design tar utgangspunkt i Tietoevrys designmanual, der blant annet skrifttype, typografiske nivåer, farger, og form er gjengitt i produktet. Det grafiske oppsettet ble også utformet i henhold til prinsippene innen calm technology, der resultat fra brukertestene tyder på at produktet var tilstrekkelig innenfor dette (vedlegg 5 Oversikt over brukertester).

Reliability

Produktet ble ikke satt i produksjon under systemutviklingen, og hadde dermed heller ikke nedetid.

Performance

Produktet er avhengig av å hente ut data fra en eller flere PDF-filer, men bruker ikke uforholdsmessig lang tid på å returnere disse dataene.

Supportability

Visjonsdokumentet, se vedlegg 2, setter krav om tilstrekkelig testing, samt en testbar kodebase. Produktet har blitt testet med en kombinasjon av *unit*- og integrasjonstester, og har en dekningsgrad på 41,9%. Ved å bruke teknikken *dependency injection* i *backend*, vil koden være lettere testbar i fremtiden siden avhengigheter kan byttes ut med etterligningsmodeller. Det tillater også lettere utbytting av avhengigheter.

Testene er beskrevet i utfyllende detalj i vedlegg 6 Systemdokumentasjon.

4.3 Administrative resultater

De administrative resultatene til prosjektet presenteres i dette kapittelet. Det omfatter timeregnskap, fremdriftsplan og dokumentasjon av systemutviklingsprosessen.

4.3.1 Timeregnskap

I løpet av prosjektperioden dokumenterte gruppen arbeidet i timelister og ukesrapporter (Vedlegg 1 Prosjekthåndbok). I figur 4.4 sees totalt antall timer brukt under prosjektperioden for hvert enkelt medlem av gruppen.

Tabell 4.4 Totale timeantall for gruppen

Navn	Timer
Linda K. Larsen	515
Jenny F. Blindheimsvik	494
Nora E. Jansrud	502

4.3.2 Fremdriftsplan

I starten av prosjektperioden ble det utarbeidet en forprosjektplan med fremdriftsplan i form av et GANTT-diagram Vedlegg 1 Prosjekthåndbok). I diagrammet listes alle planlagte aktiviteter i fire kategorier; forarbeid, produktutvikling, administrativt arbeid og avsluttende arbeid. Hver aktivitet har tilhørende estimert timer og hvilke uker de skal gjennomføres.

I tabell 4.5 sees planlagte timer opp mot faktiske timer brukt på de ulike aktivitetene. I tabell 4.6 sees planlagte timer for hver Sprint opp mot faktiske timer brukt. Her omfatter planlagte timer alt arbeid som skulle gjøres innenfor tidsrammen til Sprinten.

Tabell 4.5 Planlagte timer sammenliknet med førte timer

Oppgave	Sum timer	Planlagt	Differanse fra Gantt
Forprosjektplan	74	70	−4
Visjonsdokument	53	50	−3
Kravdokumentasjon	24	39	15
Prototyping	51	80	29
Produktutvikling	657	785	128
Brukertesting	37	26	−11
Møter	90	114	25
Informasjonsinnhenting	87	45	−42
Hovedrapport	307	240	−67
Prosjekthåndbok	24	45	22
Systemdokumentasjon	44	45	1
Presentasjon av plakat	36	30	−6
Forberedelser til presentasjon av bachelor	30	30	0
Totalt	**1 511**	1599	88

Tabell 4.6 Fordeling av timer på hver sprint

Sprint	Uker	Dager	Sum timer	Planlagt	Differanse fra Gantt
1	6-8	6	118,0	157,6	39,6
2	9-11	6	119,0	157,6	38,6
3	12-13	7	137,5	186,7	49,2
4	14-16	10	266,0	231,0	−35,0
5	17-18	10	309,5	231,0	−78,5
		39	950,0	963,7	13,7

4.3.3 Systemutviklingsprosess

Under systemutviklings perioden benyttet gruppen Scrum med artefaktene produkt *backlog*, Sprintbacklog og *burndown chart,* samt aktivitetene Sprint planleggingsmøte, daglig Scrummøte, Sprint retrospektiv og Sprintreview. Azure DevOps ble brukt til organisering av prosessen.

Ved prosjektstart ble det utarbeidet en produkt-*backlog* med prioritert rekkefølge etter estimert tid og verdi i Azure Boards. For hver Sprint, er Sprint planleggingen med oppsatte hovedmål, samt undermål, dokumentert i ukesrapportene (vedlegg 1 Prosjekthåndbok).

Hovedmålene for hver Sprint var:

Sprint 1: Laste opp en CV og få opp informasjon fra denne.

Sprint 2: A til B, fra innsending av CV til visualisering av kompetanse.

Sprint 3: Databehandling og visualisering.

Sprint 4: Visualisere roller og kompetanseområder til en person.

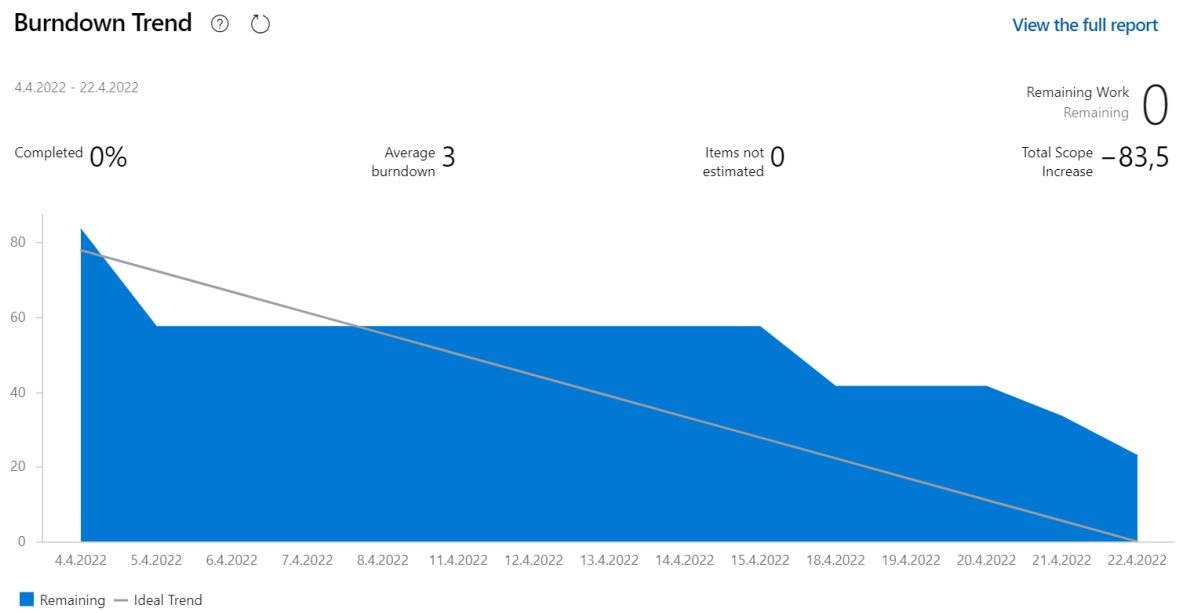
Sprint 5: Visualiser teknologier, metoder, verktøy, referanser og roller for en person og team.

Av disse ansees hovedmålene i Sprint 2, 3 og 5 som oppnådd. Målene for Sprint 1 og 4 ble tatt med videre inn i påfølgende Sprint og er derfor å anse som oppnådd i etterkant.

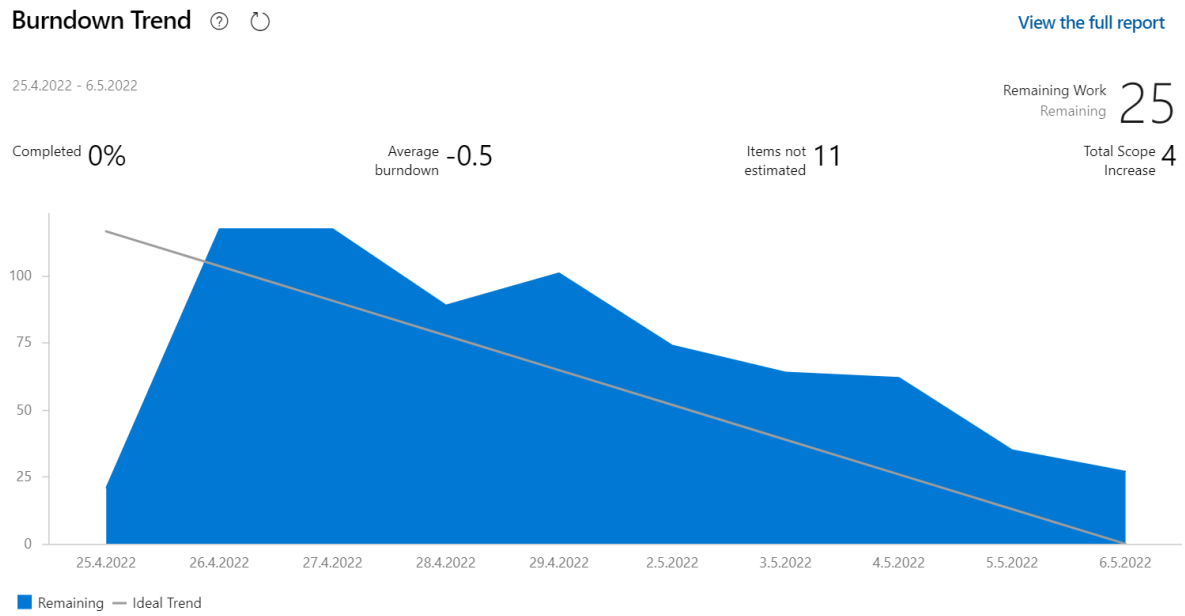
Under Sprintene ble arbeidet strukturert og organisert i Sprint tavle i Azure Devops. For Sprint 4 og 5 ble oppgaver merket med gjenværende tid, som genererte *burndown chart* fortløpende. *Burndown charts* er lagt ved i figur 4.6 og 4.7.

Etter hver Sprint ble det gjennomført Sprint retrospektiv. Refleksjoner og tiltak etter retrospektivene er dokumentert i ukesrapportene vedlagt i vedlegg 1 Prosjekthåndbok.

Sprintreview ble satt opp sammen med oppdragsgiver. Oppsummeringene av møtene er beskrevet nærmere i møtereferatene (vedlegg 1 Prosjekthåndbok).



Figur 4.7 *Burndown chart for Sprint 4.*



Figur 4.8 *Burndown chart for Sprint 5.*

Kapittel 5 Diskusjon

5.1 Vitenskapelige resultater

5.1.1 Kategorisering av data

De standardiserte CV-ene inneholder store mengder informasjon, som sammen skaper et helhetsinntrykk av den ansattes kompetanse, se vedlegg 8 Testdata. For å gi brukeren en rask oversikt over denne kompetansen gjennom datavisualiseringer, var det nødvendig å dele opp dataen i kategorier.

Kurs/sertifikater, utdanning, og prosjektbeskrivelser trekkes frem som potensielle kategorier basert på datagrunnlaget i CV-ene. Det ble konkludert med at utdanning ikke gir et godt datagrunnlag for ansatte som har tilegnet seg kompetanse fra arbeidslivet, og dermed ble ikke denne informasjonen brukt. En svakhet ved dette er at nyutdannede ansatte ikke vist hva de kan, fordi utdanning ikke er inkludert i datagrunnlaget for visualisering. Kurs og sertifikater ble nedprioritert fordi denne informasjonen er synlig og lett tilgjengelig i CV-ene. Dermed konkluderte gruppen med at det var viktigere å systematisere og visualisere kompetanseområder. Denne informasjonen oppleves som utilgjengelig for leseren, da den er flettet inn i prosjektbeskrivelser som i mange tilfeller består av lange avsnitt.

Behovet for systematisering av datasettet kom av at store datasett kan gjøre at brukeren trenger lengre tid for å tolke visualiseringen. Et mindre datasett, hvor alle datapunktene ligger innenfor samme kategori, vil resultere i visualiseringer som er lettere for brukeren å tolke.

Kompetanseområdene er kategorisert på tre forskjellige måter:

1. Hvem dataen tilhører; hvorvidt det er data tilhørende en ansatt eller et helt team
2. Hvilken type dataen har, kategorisert i roller, teknologi, verktøy, metoder eller referanser
3. Hvilken dimensjon dataen har; hvorvidt dataene for team er 2D eller 3D.

Dataen er kategorisert på denne måten for å gi brukeren en optimal brukeropplevelse. Fordelen med å skille mellom team og ansatt er tett knyttet opp mot flyten i produktet. Her var ønsket et tydelig skille, for å gjøre det tydelig for brukeren om visualiseringen representerte en enkeltperson eller et team.

Ved å kategorisere datatypene ble datagrunnlaget fra CV-ene fragmentert til fem kategorier:

- Roller
- Metoder
- Teknologier
- Verktøy
- Referanser

Det hadde vært mulig å kun benytte tre av disse; roller, kompetanseområder og referanser, da flere kategorier skaper merarbeid i forhold til produktutvikling. Likevel ble det prioritert å bruke alle fem, slik at brukeren har mulighet til å vurdere de ulike kompetanseområdene uavhengig av hverandre. Utover dette hadde vært hensiktsmessig å kunne fragmentere dataen innen hver kategori ytterligere, for å sortere dataene i

underkategorier. Ved å bruke dette i visualiseringene kan brukeren lese av kompetansen på et mindre detaljnivå, som gir mulighet for en overordnet oversikt over dataen i hver kategori.

Å bruke datainput med forskjellige dimensjoner er viktig for å bestemme hvilket detaljnivå man ønsker å visualisere dataene på. For en ansatt er alt datagrunnlag i 2D. For team skilles det mellom data i 2D og data i 3D, der data i 2D er den samlede dataen for hele teamet innenfor en kategori. Det ble vurdert som hensiktsmessig å strukturere dataene på denne måten for å kunne visualisere data for team på ulike detaljnivåer.

5.1.2 Visualisering av data

For at produktet skal utøve sin funksjon, er det viktig av datavisualiseringene er gode. Derfor ble det gjennomført en undersøkelse i form av brukertester for å samle informasjon om hvordan personer tolker forskjellige visualiseringer. Resultatene som henvises til her er presentert i tabell 4.1 og 4.2. Fremgangsmåten for å hente inn disse resultatene er beskrevet i kapittel 3.1.2 Kvalitative intervju og brukertester, samt i vedlegg 5 Oversikt over brukertester.

I utgangspunktet var hensikten med å samle inn data om visualiseringer å finne noen gode svar på hvordan det er best å visualisere data, samt å samle inn kvantitative data som kan brukes senere i rapporten. Tanken bak dette var at mye av datagrunnlaget for denne rapporten baserer seg på kvalitative data. Som forklart i kapittel 2.5 er kvalitative data gjerne farget av meningen til den som samler inn dataene, og kan dermed ha et subjektivt preg. Derfor ble det vurdert at det var et behov for noe kvantitative data for å gjøre datagrunnlaget noe mer objektivt.

Ideelt sett burde det blitt samlet inn data fra flere enn 11 personer for å kunne konkludere på denne dataen. Grunnen til at dette ikke ble gjort var at det ble vurdert som nødvendig å ikke bare vite hvilke diagrammer som var best likt, men også å samle inn data om testpersonenes tolkning av de forskjellige diagrammene. Det ble prioritert å gjennomføre en kvalitativ studie i begynnelsen av utviklingsfasen for å legge grunnlaget for utviklingen. Når produktet har nok funksjonalitet til at det er mulig å gjennomføre kvantitative studier, ville dette være et naturlig neste steg i datainnsamlingsfasen.

En annen svakhet ved datainnsamlingen er at man kun har visualisert ved hjelp av diagrammer. Visualisering av data er ikke begrenset til diagrammer, og i mange tilfeller kan en tabell eller kun et enkelt tall være det beste alternativet for visualisering. Derfor burde brukertesten på visualisering også ha omfattet tabeller og vanlig tekst.

Ut fra resultatene på undersøkelsen om diagrammer for team kommer det tydelig frem at det desidert mest populære diagrammet var det stablede søylediagrammet, som viste data i 3D. Dette var det eneste diagrammet som viste 3D-data i undersøkelsen. Dette er en svakhet, da datagrunnlaget for dette diagrammet og diagrammene det ble sammenliknet med, ikke var likt. Dermed kan det være vanskelig å vite om dette er en god måte å visualisere 3D-data på, da det ikke er noe sammenligningsgrunnlag. En fordel ved å sammenlikne diagrammer med 2D- og 3D-data som datagrunnlag er at det kommer tydelig frem et ønske blant testpersonene om å vise data med høyere detaljnivå enn 2D.

Fordelene ved å gjennomføre brukertestene slik de ble gjennomført, er at det ble samlet inn mye data om hvordan andre mennesker tolker ulike diagrammer. Disse dataene hadde stor betydning for hvordan produktet ble utviklet videre. Der tanken originalt var å

bruke et diagram per visualisering, ga resultatene fra diagram-undersøkelsen en indikasjon på at hva slags diagram man ønsker å benytte seg av, varierer fra person til person, og at det også er situasjonsbetinget. Derfor ble kursen endret, og man endte opp med å gi brukeren mulighet til å selv velge hva slags diagram som skal visualiseres. Informasjonen om at dette er situasjonsbetinget, og også sterkt knyttet opp mot tidligere erfaring, hadde ikke vært tilgjengelig om man kun hadde kjørt en kvantitativ innsamling av data da testpersonens tolkninger og reaksjoner ikke hadde blitt loggført.

5.1.3 Fasilitering av prosess for teamsammensetting

Gjennom brukertester og kvalitative intervjuer har det kommet frem mange gode forutsetninger som danner grunnlaget for hvordan man skal kunne fasilitere prosessen med teamsammensetting ved hjelp av et verktøy for visualisering av kompetanse.

Kvalitative intervjuer, i kombinasjon med brukertester, ble valgt som metode for å samle inn data om prosess med teamsammensetting, samt avdekke områder med forbedringspotensialer ved denne. Fordelen med å benytte seg av kvalitative intervju for å samle inn data om prosess er at man samler inn tolkninger og vurderinger som gjøres av mennesker med relevant arbeidserfaring. Ulempene er at man setter begrensninger på hvor mye data det er mulig å samle inn da kvalitative intervjuer er en tidkrevende prosess. Dette fører til at sammenligningsgrunnlaget blir mindre.

En annen ulempe med dataene samlet inn fra brukertestene er at de kun er samlet inn for prototypene, og ikke selve applikasjonen. Dette kom av at applikasjonen ikke var ferdig på et tidlig nok tidspunkt til at det var mulig å gjennomføre brukertester på den. En forbedring til dette ville vært å ferdigstille applikasjonen på et tidligere tidspunkt slik at det hadde vært mulig å gjennomføre brukertester på den iterativt ved implementasjon av ny funksjonalitet. Årsaken til at dette ikke ble gjennomført i dette prosjektet var at det ble prioritert å bruke mye tid på å forstå eget datagrunnlag, samt kategorisering og datavisualisering.

Et viktig perspektiv å ha med seg når man skal vurdere om produktet kan bidra til å fasilitere prosessen med å sette sammen team, er hvorvidt en CV kan vise alt som er viktig for en god prosess. Altså hvorvidt datagrunnlaget er tilstrekkelig. Det er to mangler ved datagrunnlaget som er verdt å merke seg. For det første inneholder ikke CV-ene informasjon om hvem som jobber godt sammen. Videre blir ikke *soft skills*, som innebærer oppførsels- og kommunikasjonsferdigheter, en person har fremhevet.

Den andre mangelen handler om hvordan erfaring blir målt. Ved innleveringstidspunktet vektes kompetanseområder etter hvor mange måneder man har brukt på å jobbe med prosjekter som inneholder disse kompetanseområdene. En svakhet ved dette er at kompetanseområdene ikke vektes etter hvor mange prosent stilling man har hatt på dette prosjektet, eller hvor mye man har jobbet med det. En person kan ha jobbet på tre forskjellige prosjekter i tre måneder, med fordelingen 60%, 20% og 20%. Alle disse prosjektene vil vektes like mye i vår fremstilling. Dette er en svakhet ved datagrunnlaget, da det ikke finnes informasjon i CV-ene om hvor stor stillingsprosent de forskjellige oppdragene har hatt.

5.2 Ingeniørfaglige resultater

5.2.1 Funksjonelle egenskaper

Gjennom dialog med Produkteier tidlig i utviklingsprosessen ble det fastslått at hovedprioriteten var gode visualiseringer. Denne prioriteringen underbygges i resultat- og effektmålene satt av oppdragsgiver selv, og reflekteres i *user storiene* (vedlegg 4, Powerpoint fra oppstartsmøte, Vedlegg 3 Kravdokumentasjon). Produktets grad av måloppnåelse vurderes på bakgrunn av dette, da dette gir et mer korrekt bilde av prosjektet omfang enn listen over funksjonelle krav i vedlegg 2 Visjonsdokument.

Utviklingsprosessen resulterte i et system med forholdsvis enkel funksjonalitet; en bruker kan laste opp en CV, enten for person eller for et helt team, visualisere innholdet grafisk, samt laste ned disse visualiseringene. Kjerneaktiviteten i denne prosessen har i høy grad bestått av arbeid mot å nå resultat- og effektmålene. Som en følge av dette har mange av de funksjonelle kravene spesifisert i vedlegg 2 Visjonsdokument blitt nedprioritert.

Som beskrevet i kapittel 5.1, viste resultatet av brukertestene at det var behov for individuelle valg for visualisering. Basert på dette ble det prioritert å implementere flere diagrammer for datavisualisering fremfor annen funksjonalitet, for å gi brukeren valgfrihet. Disse diagrammene ble utformet ved bruk av D3.js. Dette biblioteket har støtte for implementasjon av alle nødvendige diagrammer, med der andre rammeverk fungerer rett ut av boksen krevde D3.js en høyere grad av manuell tilpasning. Implementering av flere diagrammer førte dermed til en stor tidsmessig kostnad. Det ble likevel vurdert at tilpasningsmulighetene som følger med D3.js var verdt kostnaden, spesielt med tanke på videre utvikling av produktet.

Til tross for innsatsen mot visualisering, ble ikke alle *user storiene* som omhandlet visualisering 100% fullført. Akseptansekravene knyttet til disse *user storiene* krevde at kompetansen ble visualisert i underkategorier. Oppdragsgiver fremmet også et ønske om dette i møte 5. mai 2022. Ved å bruke underkategorier ville dataene fått en ekstra kontekst, og dermed gitt brukeren enda et verktøy til å lese av kompetansenivået til en ansatt. Omfanget av å implementere denne funksjonaliteten var for stort til at det var gjennomførbart innenfor den gitte tiden. Det var dermed ikke grunnlag for å prioritere dette akseptansekravet.

Produktet gir brukeren mulighet til å laste ned hvert enkelt diagram som vektorbasert grafikk (SVG), men har ikke støtte for å laste ned flere diagrammer i samme fil. Nedlastning av flere diagrammer samtidig kunne hatt stor nytte for brukeren som da hadde hatt mulighet til å visualisere to kompetanseområder samtidig og sammenligne disse opp mot hverandre. Støtte for nedlastning av diagrammene ble implementert mot slutten av Sprint 5. På dette tidspunktet var det ikke nok gjenstående tid til å implementere funksjonalitet som lot brukeren laste ned flere diagrammer samtidig.

De to lavest prioriterte *user storiene* hadde 0% måloppnåelse, og ble ikke påbegynt under utviklingsprosessen. Akseptansekriteriene gikk ut på å implementere et brukersystem og database for utfyllende funksjonalitet som tillater brukeren å lagre ansatte og sette sammen team basert på disse lagrede personene. Lagring av personlig informasjon krever at produktet oppfyller kravene til GDPR, som fører til en større arbeidsmengde under implementering av denne funksjonaliteten. Et brukersystem vil ha ytterligere krav i form av passordhåndtering og sikkerhet rundt dette. Støtte for denne

funksjonaliteten er ikke nødvendig for å oppfylle resultat- og effektmål, men resultatet fra flere brukertester (se vedlegg 5 Oversikt over brukertester) viser til at slike støttefunksjoner hadde gitt brukeren flere verktøy under arbeidet med å sette sammen team. Den store arbeidsmengden det ville krevd å implementere et slikt system ville tatt ressurser fra arbeidet med gode visualiseringer, og ble dermed ikke prioritert.

5.2.2 Ikke-funksjonelle egenskaper

Functionality

Ved å ikke implementere et brukersystem og lagring av CV, falt mange av sikkerhetsaspektene bort. Det mest kritiske punktet ligger i opplastning av CV, da filopplastning er en kjent sårbarhet som kan utnyttes på flere måter. Brukeren gis minimal sjanse til å laste opp skadelige filer ved å begrense opplastingen til en fil. ASP.NET setter også en øvre grense (4 MB) for størrelsen på innsendt fil.

UX - user experience

Ved å strebe mot å oppnå minst AA-sertifisering fra WCAG-standarden er det en kvalitetssikring på brukervennlighet, ikke minst i forhold til brukere med ulike helseutfordringer.

Bruk av de grafiske retningslinjene gitt av Tietoevry gjorde at produktet kan sees på som en tilhørende del av selskapet. Disse retningslinjene er utarbeidet av profesjonelle designere, og det ligger mye hjelp i å emulere arbeidet de allerede har gjort. Det var uten tvil et tidsbesparende element, da det ikke var behov for å fokusere på hvilke farger, skrifttyper og former elementene i produktet skulle ha da dette allerede var gjort for oss.

Brukertestene ga gode tilbakemeldinger på hvorvidt flyten i produktet var logisk og lett forståelig. Det hadde vært en fordel å brukerteste mer på det faktiske produktet og ikke bare en lo-fi prototyper, selv om produktet er basert på disse prototypene. Det kan tenkes at dette hadde ført til flere gode tilbakemeldinger som gikk helt konkret på løsningen. Løsningen ble bevisst utviklet for å holde forstyrrende elementer til et minimum. Spesielt knappene som var nødvendig i forhold til visualisering var en utfordring i forhold til brukervennlighet, der tilbakemeldinger fra brukertestene ble brukt for å angi form og plassering på disse.

Reliability

Azure Devops har god støtte for CI/CD og dermed gode muligheter for implementasjon av dette, men tester ble implementert sent i prosessen og behovet meldte seg dermed ikke før mot slutten av utviklingsprosessen. På dette tidspunktet ble det vurdert at ressursene var bedre brukt på andre områder, derfor ble ikke CI/CD implementert. Dersom produktet hadde blitt satt i produksjon etter første iterasjon, ville oppdragsgiver kunne gitt mer rullerende tilbakemeldinger under iterasjonene, noe som helt klart ville vært nyttig for prosessen.

Performance

Selv om det ble fastslått at hastigheten til produktet hadde lav prioritet, ble produktet utformet slik at det hadde gode forutsetninger til å svare raskt der det var

hensiktsmessig og ikke krevde mye tid. Ved å kun sende forespørsel til backend når ressursen blir etterspurt av brukeren, unngår man at det går ekstra tid på å laste inn ubrukte ressurser. Dette var spesielt viktig for metodene som henter sortert data, da disse har en tidskompleksitet på $O(N^2)$.

Generelt belager logikken som henter ut data seg mye på løkker. Dette bidrar til å øke tidskompleksiteten, og ble tidlig et punkt for bekymring. Et alternativ for å løse dette hadde vært å bruke maskinlæringsteknikker for å hente ut data, men dette lå utenfor oppgavens omfang. Et annet alternativ som ble utforsket var et bibliotek som kunne lese data basert på en fastsatt mal, men dette verktøyet var ikke fleksibelt nok til å kunne tilpasses CV-ene.

Supportability

Tietoevry har engelsk som universelt fagspråk. All kildekode er derfor dokumentert og skrevet på engelsk. I selve brukergrensesnittet ble det valgt norsk siden datagrunnlaget ville i stor grad være på norsk. Å legge inn funksjonalitet for å oversette CV-ene ville tatt uforholdsmessig lang tid. Man kunne riktignok benyttet engelske CV-er, men alle involverte parter var komfortable med norsk som språk. Det ble sett på som mest hensiktsmessig å fokusere på funksjonalitet fremfor språk i applikasjonen.

Testdekningen til applikasjonen *backend* endte på 47% som nevnt i 4.2.2 Ikke-funksjonelle krav. Denne kunne med fordel vært høyere. Mye ny funksjonalitet ble implementert siste periode av produktutviklingen. Det er i stor grad denne funksjonaliteten som behøver flere tester. Den viktigste funksjonaliteten til systemet, som opplastning, samt prosessering av innhold, blir testet. Videre, ved å gjennomgående bruke *dependency injection,* er koden letter testbar i fremtiden.

5.3 Administrative resultater

5.3.1 Timeregnskap

Det kommer frem av timelistene, vedlagt i vedlegg 1 Prosjekthåndbok, at en stor andel av timene ble jobbet i løpet av prosjektets siste del. I forhold til fordelingen av timer mellom de ulike Sprintene, se tabell 4.8, ga dette et uheldig utslag hvor de siste to Sprintene ble mye lenger enn de tre første. En jevnere timefordeling hadde vært heldigere med tanke på beregning av hvor mange oppgaver som ble fullført i løpet av en Sprint. En av årsakene til den ujevne timefordelingen var at gruppen hadde ujevn fordeling av dager mellom Sprintene. Dette kom av at gruppen jobbet med et annet fag i starten av prosjektet, og prøvde å kombinere relativt like lengder på Sprintene med at det ikke skulle bli for lenge mellom hver oppstart av en Sprint. En annen årsak til denne ujevne timefordelingen var i hovedsak uforutsette hendelser utenfor gruppens kontroll, som sykdom.

Etter uke 13 var det mulig for gruppen å jobbe fulle uker med prosjektet, da det ikke lenger var to fag på timeplanen. Dette sees tydelig på timelistene (vedlegg 1 Prosjekthåndbok), og særlig på figur 4.5, at arbeidsmengden tok seg betraktelig opp

etter påske i de to siste Sprintene. Det var også et mål for gruppen å nå 500 timer innen innleveringsfristen.

5.3.2 Fremdriftsplan

I all hovedsak følges fremdriftsplanen med enkelte avvik fra estimert start, slutt og antall timer.

Aktiviteter med spesielt avvik eller bemerkelser:

- **Kravdokumentasjonen** krevde et lavere antall timer, men ble skrevet over en lengre periode enn antatt. Dette kan sees i sammenheng med at prototypingen kom i gang senere enn først antatt og ble gjennomført iterativ.
- **Prototyping** ble ikke gjennomført i uke 3-4 men iterativt gjennom hele systemutviklingsprosessen. Dette kunne med fordel vært startet på tidligere, særlig utformingen av ulike visualiseringer til brukertesting.
- **Produktutvikling** tok mindre tid enn planlagt. Timer brukt på å sette seg inn i teknologiene som skulle brukes ble i mange tilfeller ført på informasjonsinnhenting i stedet for produktutvikling. Videre ble møter knyttet til Scrum og utviklingen, som Sprint retrospektiv, planlegging, og i noen tilfeller daglig Scrummøte, ført på møter.
- Det ble brukt mer tid på **brukertester** enn først antatt. Det ble sett på som hensiktsmessig å få god tilbakemelding fra brukergruppen, særlig med tanke på det vitenskapelige aspektet ved oppgaven. Selve gjennomføringen av testene tok også lenger tid enn først antatt. Gruppen fikk gode innspill fra UX designere fra Tietoevry på gjennomføring og innhold på brukertester. En god del av brukertestene kunne med fordel vært gjennomført tidligere i prosessen.
- **Møter** gikk under antall planlagte timer. Dette skyldes til dels at oppdragsgiver hadde begrenset tilgjengelighet og møtene med produkteier dermed ble gjennomført mer effektivt. Videre ble noen av møtene avlyst av diverse årsaker.
- **Informasjonsinnhenting** står i timelistene, men skulle i realiteten vært merket som *hovedrapport teoridel* som i GANTT-diagrammet (Vedlegg 1 Prosjekthåndbok). En konsekvens av dette er at flere timer på læring og produktutvikling er ført her.
- Det ble brukt mer timer på **hovedrapport** enn først antatt. Det var kun estimert å arbeide med hovedrapporten fra uke 18 til 20. Arbeidet med teoridel og metode ble startet før dette. Dette så gruppen på som hensiktsmessig for å legge et godt grunnlag for utviklingsarbeidet i de siste Sprintene.
- **Prosjekthåndbok** tok mindre tid enn først antatt. Noe av årsaken til dette er at timeantallet ikke reflekterer realiteten. Det er ført svært få timer på denne aktiviteten, til tross for at timelister og ukesrapporter ble oppdatert hver uke.

Resterende aktiviteter holdt seg innenfor planlagt tidsramme og varighet.

I ettertid ser gruppen at det kunne vært hensiktsmessig å ha satt av en egen aktivitet til databehandling og kategorisering. Arbeidet med analysering av datagrunnlaget og hvordan dataene skulle behandles skjedde på ulike måter og var vanskelig å plassere inn i timelistene. Dette arbeidet var svært viktig både for prosess og sluttresultat, og det hadde derfor vært ønskelig om timelistene i større grad reflekterte arbeidet som ble lagt ned i denne delen. Tiden brukt på dette har derfor blitt fordelt på prototyping, produktutvikling og møter.

Videre fikk gruppen tilgang til datagrunnlaget til applikasjonen sent. De første eksempel-CV-ene fikk gruppen tilgang til i uke 11, og hele portalen i uke 15. Arbeidet med datamodellering og kategorisering er sterkt knyttet til CV-ene, og kunne derfor ikke fullføres uten datagrunnlaget. Deler av arbeidet ble satt i gang før tilgangen til CV-ene ordnet seg, men arbeidet kom ikke i gang for fullt før uke 11.

5.3.3 Systemutviklingsprosess

Scrum ga gruppen gode rammer for produktutviklingsprosessen. Dedikert tid til planlegging, gjennomføring, evaluering og omprioritering underveis i prosessen ga gruppen et godt grunnlag for å gjennomføre prosessen på en planlagt og gjennomtenkt måte. Produkt *backlog*, Sprint planleggingsmøte, daglig Scrummøte og retrospektiv, samt Sprintreview satte goder rammer for Sprintene. Produkt *backloggen* som ble utarbeidet i starten av prosjektet var svært omfattende, og ikke formulert med fokus på bruksverdien. I løpet av Sprint 2 ble derfor elementene omformulert til å reflektere *user stories* utformet i kravdokumentasjonen.

Azure Devops var et godt og nyttig verktøy for samarbeid rundt kodebasen og struktureringen av selve Sprintene. Dette ga gruppen mulighet til å holde oversikt på arbeidet, og forhindret at flere begynte på samme oppgave. Samtidig gjorde det at gruppen enkelt kunne prioritere rekkefølgen på oppgavene. En av ulempene med Azure Devops var at gruppen ikke kjente til det fra før. Dette gjorde at en del funksjonalitet, som for eksempel automatisk generering av *burndown-charts* ble tatt i bruk sent i prosessen.

I struktureringen av arbeidet ble produkt *backlog* og Sprintbacklog med *taskboard* i Azure Board aktivt benyttet. I de to siste Sprintene ble utestående tid benyttet på oppgavene, som igjen automatisk genererte *burndown charts* som vist i figurene 4.6 og 4.7. Denne funksjonaliteten burde vært benyttet fra første Sprint. Å sette en tidsbegrensning på oppgaver ga gruppen et estimat på hvor mye tid en oppgave skulle ta. Dette ga også dermed en indirekte prioritet av oppgavene, samt en tidsfrist for når de skulle være fullført.

Azure Repo gjorde versjonskontroll av systemet lettvint. *Branch*-strategien som ble det benyttet var *trunk-based* strategi. Fordelene ved å benytte en slik strategi var at *master branch* frekvent ble oppdatert med ny kode. Det gjorde det enkelt for alle i gruppen å holde seg oppdatert på arbeidet. Ulempene med en slik strategi var at det frekvent ble merget kode til master. Dette medfører en økt risiko for å merge små feil og kode som krasjer programmet inn i master. Mot slutten av utviklingsfasen ble det implementert mye ny funksjonalitet. Dette medførte en stor jobb med å rydde opp i feil mot slutten av utviklingsfasen.

Hovedmålet for Sprint 1 og 4 ble ikke nådd. For Sprint 1 skyldtes det i stor grad at det gikk med mer tid enn beregnet til oppsett av systemet, samt å sette seg inn ny teknologi. Utydelige mål med vage akseptansekrav var også noe av årsaken til dette. Videre var det behov for mer arbeid med dataene for å legge et godt grunnlag for uthenting og visualisering. For Sprint 4 ble ikke målene nådd grunnet problemer med implementasjon av de grafiske fremstillingene. Dette ble løst raskt i neste Sprint.

Gruppen har hatt særlig stor nytte av retrospektivene som ble gjennomført etter hver Sprint. Som resultat av disse fikk gruppen satt inn flere tiltak for å ta tak i aspekter av prosessen som ikke fungerte optimalt. Etter Sprint 3 kom for eksempel gruppen frem til

at det var et behov for å sette mer konkrete mål. Tydeligere akseptansekrav ga bedre oversikt over produktutviklingen ved å gjøre det lettere å markere en oppgave som ferdig.

Sprintreview ble gjennomført med oppdragsgiver og ga produkteier muligheten til å jevnlig gi tilbakemelding på produktet, slik at misforståelser og feilvurderinger fort kunne rettes opp i. Disse møtene burde vært lagt nærmere oppstart av neste Sprint, slik at det ble enklere å implementere endringer etter produkteiers ønske.

5.3.4 Gruppesamarbeid

Gruppen har vært tilpasningsdyktige og hatt fokus på å spille hverandre gode gjennom hele prosjektperioden. Alle i gruppen har hatt like stort ambisjonsnivå for oppgaven og stor grad tatt eierskap til den. Samlet har gruppen utfylt hverandre på forskjellige områder, og arbeidsbelastningen har vært jevn. Gjennom faglige diskusjoner har gruppen løst uenigheter på en konstruktiv måte.

En utfordring gruppen har måtte håndtere gjennom prosjektperioden er at medlemmene på gruppen er svært forskjellige. Dette gjorde at gruppen var nødt til å fokusere på god og konstruktiv kommunikasjon gjennom hele perioden. Prinsippet om at det var greit å ta opp aspekter av gruppesamarbeidet som man ikke var fornøyd med, samt krav til at disse skulle tas imot på en konstruktiv måte gjorde at gruppen klarte å håndtere disse individuelle forskjellene.

Etter samfunnets gjenåpning ble gruppen på forskjellige tidspunkter utsatt for sykdom som reduserte arbeidskapasiteten blant medlemmene. Dette reflekteres også i timelistene (Vedlegg 1 Prosjekthåndbok). Ved god og tydelig kommunikasjon har gruppen sikret videre arbeid også i disse periodene.

Gjennom prosjektperioden fikk gruppen benyttet oppdragsgivers kontorlokaler. Dette var praktisk med tanke på bruk av grupperom, møterom og arbeidsplasser, samt muligheten til å være tett på oppdragsgiver.

Utover dette er gruppen fornøyd med gjennomføring av prosessen. Særlig har den jevnlige dokumenteringen av timelister, ukesrapporter og møtereferater har vært til stor nytte for gruppen.

Kapittel 6 Konklusjon og videre arbeid

6.1 Konklusjon

Arbeidet med denne oppgaven har hatt som hensikt å svare på problemstillingen definert i kapittel 1.2:

> «Hvordan kan en webapplikasjon for visualisering av kompetanse utvikles for å fasilitere sammensetning av et profesjonelt team?»

Resultatet fra brukertestene beskrevet i kapittel 4.1 viste at kompetanse fremstilt i diagrammer har potensiale for å være til stor nytte i en slik prosess. Hva slags visualisering som er egnet for hvilke scenario avhenger av flere variabler. Brukertestene ga også en indikasjon på at når det kommer til selve utformingen av visualiseringer er det store variasjoner mellom hva forskjellige individer foretrekker. Dimensjonen på dataen, og hvilket detaljnivå man ønsker å fremstille dataene spiller også en rolle for utformingen. Det siste funnet er at formålet med visualiseringer også spiller en rolle; man ønsker å bruke forskjellige diagrammer for forskjellige formål.

Et annet aspekt som er viktig for å kunne svare på problemstillingen er hvordan dataene blir segmentert. Som poengtert i 4.1 Vitenskapelige Resultater, er det vanskelig å fremstille store datasett med mye input på en god og oversiktlig måte. Derfor ble datasettet segmentert, som presentert i kapittel 4.1 og 5.1.

Å fasilitere sammensetning av et profesjonelt team innebærer å gjøre denne prosessen enklere. Ut fra tilbakemeldingene og grunnlaget for prosessen fra Tietoevry, se vedlegg 4 Powerpoint fra oppstartsmøtet, samt tilbakemeldinger fra brukertester, finnes det forskjellige måter å fasilitere denne prosessen på. Som presentert i kapittel 4.1 og 5.1 baserer prosessen med å sette sammen team seg mye på hvem den enkelte prosjektlederen kjenner til. Dette gjør at prosessen er preget av subjektivitet. Dermed vil fasilitering av prosessen innebære å gjøre denne prosessen mer objektiv. Et annet aspekt som kom frem av brukertestene, se vedlegg 5 Oversikt over brukertester, er at det kan være svært tidkrevende å finne de riktige personene til riktig arbeid.

For å gjøre prosessen mer objektiv, ville produktet trengt noe utbedring. Videre arbeid for å oppnå dette er nærmere beskrevet i kapittel 6.2 Videre arbeid. En database hvor man lagrer alle ansatte, samt en filtrering på visse kriterier, slik at man kan finne den ansatte som passer best inn i teamet basert på kompetanse, ville gjort prosessen mindre subjektiv. Det gjør det mulig å oppdrive de ansatte man trenger uten å måtte basere seg på å spørre de man kjenner til. Dermed vil ikke nettverk i bedriften bli et kriterium, og prosessen ville blitt mer objektiv. Dette ville også bidratt til å effektivisere prosessen. Ved å fjerne behovet for å nærlese lange CV-er og sammenlikne dem, kan man enkelt se kompetansen hos ansatte og sammenlikne dem.

Produktet slik det eksisterer i dag hadde trengt utbedringer jamfør de som er beskrevet i avsnittet over for å kunne fasilitere teamsammensetningen. Likevel antyder resultatene fra brukertester at det er et behov for funksjonaliteten som er implementert i dag, som presentert i 4.1 og diskutert i 5.1.

6.2 Videre arbeid

I løpet av prosjektperioden har gruppen fått begynt på et solid fundament for produktet. En god MVP er blitt utviklet med intuitive visualiseringer. Ved å bygge videre på produktet kan man i fremtiden få et viktig verktøy i prosessen for å fasilitere for teamsammensetning. Her vil gruppen presentere de mest nærliggende forslagene til videre implementasjon og funksjonalitet for å bringe produktet til nye høyder.

Implementering av kontinuerlig utrulling, samt økt testdekning ser gruppen på som det mest nærliggende videre arbeidet om systemutviklingen skulle fortsatt. Videre ville gruppen fortsatt fokuset på user storisene i Vedlegg 3 Kravdokumentasjon, med bedre visualiseringer, lagring av ansatte, samt brukerhåndtering.

Visualiseringene i applikasjonen kunne blitt bedre dersom man fikk satt dataene i et ekstra lag med kontekst, som nevnt i kapittel 5.2. Dette vil kreve videre segmentering av data. En slik kategorisering krever semantisk forståelse av de ulike kompetanseområdene og underkategorier. Et slikt arbeid ville med fordel skje i samarbeid med aktuelle fagpersoner.

Med økt funksjonalitet vil produktet ha mer potensiale til å fasilitere prosessen ved å sette sammen team. Ved å legge til funksjonalitet for lagring av ansatte med søke- og filtreringsfunksjonalitet vil det bli lettere for prosjektledere å lete frem ansatte med relevant kompetanse. Videre hadde funksjonalitet for å dynamisk sette sammen team være nyttig.

Om produktet skal lagre informasjon om ansatte krever det, som nevnt i kapittel 5.2, ekstra hensyn i forhold til sikkerhet og personvern. Ved videre implementasjon av dette er det derfor viktig at brukerhåndtering, nødvendige personvernvurderinger og -tiltak, samt andre nødvendige sikkerhetstiltak nevnt i kapittel 2.3.1 Sikkerhet, blir implementert.

Mer langsiktig videre arbeid for produktet som har kommet frem fra brukertester er muligheten til å legge inn en «perfekt» kandidat i systemet, for å deretter filtrere ut de kandidater som matcher helt eller delvis. Her er et viktig aspekt også tilgjengelighet for den ansatte. På den måten kan en prosjektleder legge inn kriteriene som gjerne kommer fra en kunde for å så få opp kandidater som de har mulighet til å tilby. Om det ikke finnes en «perfekt» kandidat kan prosjektleder da ta en vurdering på om noen av de aktuelle kandidatene passer godt nok.

For videre implementasjon og skalering vil gruppen anbefale å se på muligheten for å velge ønsket språk i applikasjonen.

6.3 Samfunnspåvirkning

Utvikling av ny teknologi kan ha større eller mindre samfunnspåvirkning, både i positiv eller negativ retning. Samfunnspåvirkningen til produktet utviklet i denne prosjektperioden kan sees særlig i sammenheng med hvilken effekt det vil ha i tilknytning sammensetning av profesjonelle team.

6.3.1 Profesjonsetiske problemstillinger

For å komme frem til det ferdige produktet, var gruppen nødt til å objektivisere prosessen ved sammensetning av team. Dette medfører noen etiske problemstillinger, da denne prosessen innebærer å skape en objektiv fremstilling av personene involvert. Å redusere en persons kompetanse til tall som kan fremstilles i et diagram er problematisk, da semantiske ferdigheter og personlig egnethet ikke blir tatt med i regnestykket.

På en annen siden kan man argumentere for at en slik objektivisering kan skape et mer rettferdig vurderingsgrunnlag ved sammensetning av team. Et objektivt datagrunnlag kan bidra til å motvirke en prosjektleders eget bias mot minoriteter innenfor bransjen. Dersom dette fører til at de mest kvalifiserte for oppdraget blir satt sammen i et team, vil dette føre til bedre prosjektgjennomføring og ha positiv effekt for bedriften.

6.3.2 Bærekraftsvurdering

IT-produkter kan ofte ha et høyt klimaavtrykk. Dette er spesielt forbundet med server-, internett- og strømbruk (*The Problem of Power Consumption in Servers*, u.d). Til tross for at det utviklede produktet er et slikt system, vurderes klimaavtrykket til å være lite. Det er ikke et produkt som er ment for bruk av store folkemasser, men til å være et spesialverktøy spisset inn mot en svært liten brukergruppe. Produktet vil dermed være relativt sjeldent i bruk. Samtidig erstatter produktet nåværende løsning, som i store trekk også baserer seg på bruk av IT-ressurser. Per dags dato lagres heller ikke noe data permanent, og ved fremtidig implementasjon av dette vil produktets natur holde mengden data på et lavt nivå.

Kilder

A Brief History of Data Visualization (2019). Tilgjengelig på: https://www.dundas.com/resources/blogs/introduction-to-business-intelligence/brief-history-data-visualization (Hentet: 17. April 2022).

'Agile Manifesto for Software Development | Agile Alliance' (2015) Agile Alliance |, 29 June. Tilgjengelig på: https://www.agilealliance.org/agile101/the-agile-manifesto/ (Hentet: 9. mai 2022).

Andersen, E.S. (2016) *Prosjektledelse: Dette må alle ledere vite.* Oslo: NKI-forlaget.

Application Protocols · How Internet Works (u.d.). Tilgjengelig på: https://makersinstitute.gitbooks.io/how-internet-works/content/application-protocols.html (Hentet: 9. Mai 2022).

Atlassian. (2022) *Is the Agile Manifesto Still a Thing?, Atlassian.* Tilgjengelig på: https://www.atlassian.com/agile/manifesto (Hentet: 19. April 2022).

Atlassian. (2022) *Kanban vs Scrum, Atlassian.* Tilgjengelig på: https://www.atlassian.com/agile/kanban/kanban-vs-scrum (Hentet: 19. April 2022).

Atlassian. (2022) *Scrum - what it is, how it works, and why it's awesome.* Tilgjengelig på: https://www.atlassian.com/agile/scrum (Hentet: 19. April 2022).

Atlassian. (2022) *What is Agile?, Atlassian.* Tilgjengelig på: https://www.atlassian.com/agile (Hentet: 19. April 2022).

Atlassian (2022) What is version control | Atlassian Git Tutorial, Atlassian. Tilgjengelig på: https://www.atlassian.com/git/tutorials/what-is-version-control (Hentet: 9. Mai 2022).

Babich, N. (2022) Prototyping 101: The Difference between Low-Fidelity and High-Fidelity Prototypes and When to Use Each, Adobe Blog. Tilgjengelig på: https://blog.adobe.com/en/publish/2017/11/29/prototyping-difference-low-fidelity-high-fidelity-prototypes-use (Hentet: 16. Mai 2022).

Bartnes, M. and Nätt, T.H. (2022) 'HTTPS', *Store norske leksikon*. Tilgjengelig på: http://snl.no/HTTPS (Hentet: 11. Mai 2022).

Bostock, M. (u.d.) *D3.js - Data-Driven Documents.* Tilgjengelig på: https://d3js.org/ (Hentet: 18. April 2022).

Brocke, J. & Hevner, A. & Maedche, A. (2020). *Introduction to Design Science Research.* 10.1007/978-3-030-46781-4_1.

Brush, K. (2020) *What is data visualization and why is it important?, SearchBusinessAnalytics.* Tilgjengelig på: https://www.techtarget.com/searchbusinessanalytics/definition/data-visualization (Hentet: 17. April 2022).

Budiu, R. (2017) *Quantitative vs. Qualitative Usability Testing, Nielsen Norman Group.* Tilgjengelig på: https://www.nngroup.com/articles/quant-vs-qual/ (Hentet: 20. April 2022)

Byyny, R.L. (2016) *Information and cognitive overload*, s. 6.

CV Partner - For Bids and Proposals (u.d.). Tilgjengelig på: https://www.cvpartner.com/ (Hentet: 18. Mai 2022).

'Design Patterns, Set 1 (Introduction)' (2015) *GeeksforGeeks*, 6. August. Tilgjengelig på: https://www.geeksforgeeks.org/design-patterns-set-1-introduction/ (Hentet: 18. Mai 2022).

Dictionary.cambridge.org. (u.d.) *Skill.* Tilgjengelig på: https://dictionary.cambridge.org/dictionary/english/skill (Hentet 30. Mars 2022)

Dvergsdal, H. (2021) 'HTTP', *Store norske leksikon*. Tilgjengelig på: http://snl.no/HTTP (Hentet: 11. Mai 2022).

File uploads, Web Security Academy (u.d.). Tilgjengelig på: https://portswigger.net/web-security/file-upload (Hentet: 9. Mai 2022).

FURPS (2021) *Wikipedia*. Tilgjengelig på: https://en.wikipedia.org/w/index.php?title=FURPS&oldid=1050953940 (Hentet: 11. Mai 2022).

Freeman, J.V. and Julious, S.A. (u.d.) *'The Visual Display of Quantitative Information'*, s. 5.

Haddad, R. (2022) *Git Branching Strategies: GitFlow, Github Flow, Trunk Based..., Flagship.io.* Tilgjengelig på: https://www.flagship.io/git-branching-strategies/ (Hentet: 21. April 2022).

Hartson, R. and Pyla, P.S. (2012) *The UX Book: Process and Guidelines for Ensuring a Quality User Experience*. Elsevier.

Holmen, Heine Alexander (u.d.) *Kunnskap* i Store norske leksikon. Tilgjengelig på: https://snl.no/kunnskap (Hentet 30. Mars 2022)

IT Consulting (u.d.). Tilgjengelig på: https://www.consultancy.org/consulting-industry/it-consulting (Hentet: 18. Mai 2022).

Kyriakidis, A., Maniatis, K. and You, E. (2017) *'The Majesty of Vue.js 2'*, s. 30.

Larkin, K., Smith, S. and Dahler, B. (2022) *Dependency injection in ASP.NET Core*, Microsoft. Tilgjengelig på: https://docs.microsoft.com/en-us/aspnet/core/fundamentals/dependency-injection (Hentet: 18. Mai 2022).

Martinez, P. (2022) *Dependency Injection Vs Dependency Inversion Vs Inversion of Control, Let's set the Record Straight,* Medium. Tilgjengelig på: https://medium.com/ssense-tech/dependency-injection-vs-dependency-inversion-vs-inversion-of-control-lets-set-the-record-straight-5dc818dc32d1 (Hentet: 18. Mai 2022).

Merriam Webster. (2022) *Competence.* Tilgjengelig på: https://www.merriam-webster.com/dictionary/competence (Hentet: 30. Mars 2022).

Microsoft (2022) *Best practices for writing unit tests - .NET.* Tilgjengelig på: https://docs.microsoft.com/en-us/dotnet/core/testing/unit-testing-best-practices (Hentet: 1. April 2022).

Microsoft. (2022) *What is ASP.NET?, .NET*. Tilgjengelig på: https://dotnet.microsoft.com/en-us/learn/aspnet/what-is-aspnet (Hentet: 19. April 2022).

Morgan, A. (2019) *Visualizing beyond 3 Dimensions*, *Medium*. Tilgjengelig på: https://towardsdatascience.com/visualizing-beyond-3-dimensions-67531b431119 (Hentet: 16. Mai 2022).

OWASP Foundation, Open Source Foundation for Application Security (u.d.). Tilgjengelig på:: https://owasp.org/ (Hentet: 11. Mai 2022).

OWASP Top Ten Web Application Security Risks, OWASP (u.d.). Tilgjengelig på: https://owasp.org/www-project-top-ten/ (Hentet: 9. Mai 2022).

Personvernprinsippene (u.d.) *Datatilsynet.* Tilgjengelig på: https://www.datatilsynet.no/rettigheter-og-plikter/personvernprinsippene/ (Hentet: 16. Mai 2022).

PDF, iText (2018) iText 7 Core, iText PDF. iTextpdf Software. Tilgjengelig på: https://itextpdf.com/en/products/itext-7/itext-7-core (Hentet: 19. Mai 2022).

Rigby, D., Sutherland, J. and Takeuchi, H. (2016) 'Embracing Agile', *Harvard Business Review*. Tilgjengelig på: https://hbr.org/2016/05/embracing-agile (Hentet 18. Mai 2022).

Rossen, E. (2020) 'API', *Store norske leksikon*. Tilgjengelig på: http://snl.no/API (Hentet: 11. Mai 2022).

Ruparelia, N. B., (2010) *The History og Version Control, Hewlett Packard Enterprise Services.* doi: 10.1145/1668862.1668876

S, S.V. and Surendran, S. (2015) '*A Review of Various Linear and Non Linear Dimensionality Reduction Techniques'*.

Scrum Guide | Scrum Guides (2022). Tilgjengelig på: https://scrumguides.org/scrum-guide.html#scrum-events (Hentet: 9. Mai 2022).

Sheth, A., (2022) *NUnit vs. XUnit vs. MSTest: Comparing Unit Testing Frameworks In C#.* Tilgjengelig på: https://www.lambdatest.com/blog/nunit-vs-xunit-vs-mstest/ (Hentet: 19. April 2022).

Software Testing Fundamentals. (2022) *Integration Testing.* Tilgjengelig på: https://softwaretestingfundamentals.com/integration-testing/ (Hentet: 20. April 2022).

Spacey, J. (2022) *38 Types of IT Industry*, *Simplicable*. Tilgjengelig på: https://simplicable.com/new/it-industry (Hentet: 18. Mai 2022).

Stephanie (2016) *Dimensionality & High Dimensional Data: Definition, Examples, Curse of*, *Statistics How To*. Tilgjengelig på: https://www.statisticshowto.com/dimensionality/ (Hentet: 16. Mai 2022).

Store norske leksikon. (u.d.) *Erfaring.* Tilgjengelig på: https://snl.no/erfaring (Hentet 30. Mars 2022)

Store norske leksikon (2018) *Kompetanse*. Tilgjengelig på: https://snl.no/kompetanse (Hentet 13. mai 2022)

Swistak, T. (2021) *Dependency Injection, Dependency Inversion and Inversion of Control Explained (TypeScript example)*, Logic Room. Tilgjengelig på: https://www.logicroom.co/blog/dependency-injection-dependency-inversion-and-inversion-of-control-explained-theory (Hentet: 18. Mai 2022).

Tan, W., Liu, D. and Bishu, R. (2009) 'Web evaluation: Heuristic evaluation vs. user testing', *International Journal of Industrial Ergonomics*, 39(4), s. 621–627. doi:10.1016/j.ergon.2008.02.012.

The Power of Figma as a Design Tool (u.d.) *Toptal Design Blog.* Tilgjengelig på: https://www.toptal.com/designers/ui/figma-design-tool (Hentet: 19. April 2022).

The Problem of Power Consumption in Servers (2022) InfoQ. Tilgjengelig på: https://www.infoq.com/articles/power-consumption-servers/ (Hentet: 19. Mai 2022).

Thorsvik, J. and Jacobsen, D.I. (2019) *Hvordan organisasjoner fungerer.* 5. Bergen: Fagbokforlaget.

Tietoevry.com (u.d.) *Tietoevry skaper meningsfull teknologi som bidrar til å gjøre verden bedre.* Tilgjengelig på: https://www.tietoevry.com/no/om-oss/om-tietoevry/ (Hentet: 28. Januar 2022).

Tufte, E.R. (1983a) *'The Visual Display of Quantitative Information', in The Visual Display of Quantitative Information.* Second edition. Conneticut: Graphics Press LLC, s. 13.

Tufte, E.R. (1983b) *The Visual Display of Quantitative Information.* Second edition. Conneticut: Graphics Press LLC.

Tutorialspoint.com. (2022) *Unit Testing.* Tilgjengelig på: https://www.tutorialspoint.com/software_testing_dictionary/unit_testing.htm (Hentet: 20. April 2022).

Unwin, A. (2020) *'Why is Data Visualization Important? What is Important in Data Visualization?'*, Harvard Data Science Review, 2(1). doi:10.1162/99608f92.8ae4d525.

Veeraraghavan, S., (2022) *Best Programming Languages to Learn in 2022.* Tilgjengelig på: https://www.simplilearn.com/best-programming-languages-start-learning-today-article (Hentet: 19. April 2022).

WCAG 2.0-standarden, Tilsynet for universell utforming av ikt (u.d.). Tilgjengelig på: https://www.uutilsynet.no/wcag-standarden/wcag-20-standarden/86 (Hentet: 11. Mai 2022).

Web Content Accessibility Guidelines (WCAG) 2.1 (2022). Tilgjengelig på: https://www.w3.org/TR/WCAG21/ (Hentet: 16. Mai 2022).

West, D. (u.d.) *Agile Scrum Roles*, *Atlassian*. Tilgjengelig på: https://www.atlassian.com/agile/scrum/roles (Hentet: 18. Mai 2022).

'What is Agile Software Development?' (2015) Agile Alliance |, 29 June. Tilgjengelig på: https://www.agilealliance.org/agile101/ (Hentet: 9. Mai 2022).

What is Scrum? (u.d.) Scrum.org. Tilgjengelig på: https://www.scrum.org/resources/what-is-scrum (Hentet: 9. Mai 2022).

What is Software Testing and How Does it Work?, IBM (u.d.). Tilgjengelig på: https://www.ibm.com/topics/software-testing (Hentet: 9. Mai 2022).

What is Usability Testing? (and What it Isn't), Hotjar (2022). Tilgjengelig på: https://www.hotjar.com/usability-testing/ (Hentet: 18. April 2022).

Williams, C. (2007) *'Research Methods', Journal of Business & Economics Research* (JBER), 5(3). doi:10.19030/jber.v5i3.2532.

What is DevOps? (u.d.). Tilgjengelig på: https://about.gitlab.com/topics/devops (Hentet: 11. Mai 2022).

What is Web Application (Web Apps) and its Benefits (uten dato) SearchSoftwareQuality. Tilgjengelig på: https://www.techtarget.com/searchsoftwarequality/definition/Web-application-Web-app (Hentet: 1. Mai 2022).

W3C (u.d.) *About W3C.* Tilgjengelig på: https://www.w3.org/Consortium/ (Hentet: 16.mai 2022)

xUnit.net. (2022) *XUnit.* Tilgjengelig på: https://xunit.net/ (Hentet: 19. April 2022).

Figurer

dev.to, (2020) *Trunk based development*. Tilgjengelig på https://dev.to/arbitrarybytes/comparing-git-branching-strategies-dl4

Rosling, H. (2021) *How does income relate to life expectancy.* Tilgjengelig på https://www.gapminder.org/tools/#uiprojector:true;&chart-type=bubbles&url=v1

Vedlegg

Vedlegg 1: Prosjekthåndbok

Vedlegg 2: Visjonsdokument

Vedlegg 3: Kravdokumentasjon

Vedlegg 4: Power Point fra oppstartsmøtet

Vedlegg 5: Oversikt over brukertester

Vedlegg 6: Systemdokumentasjon

Vedlegg 7: WCAG oversikt

Vedlegg 8: Testdata

Vedlegg 9: Kildekode

Vedlegg 2

59

Visualisering av kompetanse
Visjonsdokument

Versjon 1.0

Revisjonshistorie

Dato	Versjon	Beskrivelse	Forfatter
20.01.2022	0.1	Begynte på dokumentet. Problem- og produktsammendrag, overordnede beskrivelse av interessenter og brukere, produktoversikt	Linda K. L, Jenny F. B, Nora E. J
21.01.2022	0.2	Revidert dokumentet etter presentasjon fra oppgavestiller Roar.	Jenny F. B.
27.01.2022	0.3	Utdypet sammendrag av produkt og problem, i tillegg til interessenter og brukere.	Nora E. Jansrud
28.01.2022	0.4	Finpusset sammendrag og skrev kap. 3	Nora E. Jansrud
03.02.2022	0.5	Utdypet brukermiljø. Lagt inn om alternative produkter. Skrev ikke funksjonelle krav.	Linda K. L, Jenny F. B, Nora E. J
04.02.2022	0.6	Utbedret og ferdigstilte dokumentet.	Nora E. Jansrud
17.02.2022	1.0	Ferdigstilling for innlevering til veileder	Nora E. Jansrud
23.02.2022	1.1	Revidere etter kommentarer fra veileder.	Jenny F. B.
03.03.2022	1.2	Reviderte etter kommentarer fra veileder	Nora E. J.
11.03.2022	2.0	Ferdigstilt etter kommentarer fra veileder	Nora. E. J.

Innholdsfortegnelse

1. Innledning

Dette dokumentet er skrevet i anledning av Bacheloroppgave IDATT2900 ved studiet Dataingeniør, NTNU, utført våren 2022. Oppgaven som skal løses heter *Visualisering av kompetanse* og er stilt av Tietoevry Norway AS, herfra referert til som Tietoevry.

Løsningen på oppgaven sikter på å gi en bedre visualisering av kompetansen til ansatte og team i bedriften Tietoevry. Tietoevry er et teknologiselskap som tilbyr IT- og produktuviklingstjenester med omtrent 24 000 ansatte som er fordelt på over 90 land (Tietoevry.com, u.d). I deres arbeid setter prosjektledere sammen team som leverer produkt og tjenester til deres kunder. Kompetansen til disse teamene og de enkelte teammedlemmene er derfor nyttig å få visualisert. Særlig under kommunikasjon med aktuelle kunder for å gi dem innsyn i hva de kjøper, men óg ved sammensetning av nye team. Det kan være vanskelig å skaffe seg en oversikt over kompetansen og erfaringen til en person, et team, eller hele bedriften.

En god ressurs for å skaffe seg oversikt over kompetansen til en person er å bruke CV-er. Den består av informasjon rundt den enkelte sin utdanning, erfaring, jobbhistorie og ferdigheter. Det varierer mye hva en CV inneholder og hvordan den er strukturert. Kvaliteten på innholdet og hvor oppdatert den er er varierer også. Standardiserte CV-er for bedrifter kan gjøre det enklere å få oversikt over dens ansattes kompetanse og å kunne se på en større mengde CV-er samt sammenligne disse.

Med oppgaven *Visualisering av kompetanse* er hensikten å skape en god ressurs for visualisering av team ved hjelp av standardiserte CV-er bedriften besitter av alle sine ansatte. Løsningen vil fungere som en nettressurs som skal ta sikte på å hjelpe prosjektledere med å effektivisere prosessen med å sette sammen gode team, samtidig som det skal være et funksjonelt verktøy i forbindelse med innsalg av team til potensielle og eksisterende kunder.

Dette dokumentet er en beskrivelse av produktet det tas sikte på å utvikle. Dokumentet består av en kort beskrivelse av problemet som har dannet behovet for produktet, og et sammendrag av produktet. Deretter kommer en beskrivelse av interessenter og brukere, samt en beskrivelse av brukermiljøet. Dette er ment for å gi en forståelse av hvordan produktet er tiltenkt å fungere og for å klargjøre hvilke behov det skal fylle.

Visjonsdokumentet inneholder også en oversikt hvor produktet sees i sammenheng med brukermiljøet og de forutsetninger som ligger til grunn for utviklingen. Deretter følger en beskrivelse av produktets funksjonelle og ikke-funksjonelle egenskaper.

Dette dokumentet er en beskrivelse av produktet slik det er tiltenkt i starten av prosjektet. Dette kan endre seg gjennom arbeidsperioden og er på ingen måte et fastsatt og endelig mål for hvordan produktet skal se ut. Ei heller må det tolkes som en begrensing på hva produktet kan bli.

2. Sammendrag problem og produkt

2.1 Problemsammendrag

I konsulentbransjen er ofte målet å selge team til kunder. Utfordringen her ligger i å gi kunden en godt bilde på hva slags kompetanse og erfaring et team sitter inne med. Kompetanse og erfaring består av mange aspekter, fra harde ferdigheter til myke ferdigheter. Erfaringen og kompetansen hos en ansatt er ofte representert i form av CVer. En CV vil gi en detaljert beskrivelse av hvert enkelt team-medlems erfaring og kompetanse.

Det kan være utfordrende å få oversikt over kompetansenivået i teamet som helhet. Når en kunde skal velge et team som kan løse deres problem, kan det også være nyttig å skaffe seg en oversikt over hvilket team som vil passe best til deres behov.

En vellykket løsning skal kunne visualisere kompetansen hos individer og team på en måte som gjør at prosjektleder for god innsikt, og mer effektivt kan sette sammen gode team. Samtidig bør en vellykket løsning gjøre det enklere å gi kunden et godt bilde på teamet de velger, slik at det blir enklere å selge riktig team til riktig prosjekt.

2.2 Produktsammendrag

Produktet skal utvikles for Tietoevry Norway AS for konsulentvirksomheten deres innenfor IT. Tietoevry ønsker å få oversikt over ansattes og teams kompetanse. Løsningen vil hente ut kompetanse og erfaring hos ansatte fra standardiserte CVer, visualisere kompetanse og erfaring hos ansatte, samt hos team som helhet. Dette skal gjøre det enklere å sette sammen gode team.

I motsetning til dagens system, som baserer seg på å manuelt lese CV-er og sette sammen team basert på den subjektive oppfatningen til prosjektleder, vil denne løsningen gi muligheten til å ta mer objektive beslutninger for teamets beste. Det vil også synliggjøre eventuell mangel av kompetanse i bedriften og teamene. Videre vil det gjøre det lettere å sette sammen team med teammedlemmer basert i f.eks. ulike land, hvor en prosjektleder ikke nødvendigvis kjenner de aktuelle kandidatene.

Med dagens system må man manuelt lage diagrammer dersom man ønsker å visuelt representere et team for en kunde ved innsalg. Produktet vil kunne effektivisere denne prosessen ved at visualiseringen skjer automatisk.

I motsetning til konkurrenters løsninger vil vårt produkt ha funksjonalitet til å håndtere standardiserte CV-er fra systemet CV-partner som Tietoevry benytter seg av. Dette kan effektivisere prosessen for prosjektledelsen og bedriften ved å enkelt og raskt visualisere dataene representert i en CV.

3. Overordnet beskrivelse av interessenter og brukere

3.1 Oppsummering interessenter

3.1.1 Utviklerne

Det er tre utviklere som jobber på prosjektet.

Jenny F. Blindheimsvik fungerer som SCRUM master og har ansvaret for prosjektfremdrift og utvikling, samt for at gruppen forholder seg til innleveringsfrister og overholder planlagt tidsplan.

Linda Larsen har ansvar for versjonskontroll og kvalitetssikring av kode.

Nora E Jansrud har ansvaret for kommunikasjon med produkteier og ansvaret for å kalle inn til og lede møter.

Gruppen har fordelt rollene og oppgavene etter hvilke styrker og erfaring medlemmene har fra tidligere prosjektarbeid. Det ble sett på som hensiktsmessig da gruppen ønsker gode prosesser, kvalitetssikring og kommunikasjon. Gruppen kom frem til denne fordelingen gjennom et møte hvor man diskuterte hverandres styrker, svakheter og erfaring.

3.1.2 Produkteier

Roar S. Gjøvaag fungerer som produkteier på vegne av Tietoevry. De ønsker i sin virksomhet å sette sammen team for ulike prosjekter, og selge team, som beskrevet i kapittel 2. Produkteier skal sørge for at gruppen utvikler det produktet som er tiltenkt, og bidra til å klargjøre intensjonen i oppgaven.

3.1.3 Veileder

Elise Klæbo Vonstad er veileder, og representerer NTNU i prosjektet. Bacheloroppgaven skrives i forbindelse med studiet Dataingeniør ved Institutt for datateknologi og informatikk ved NTNU. Veileder skal bistå gruppen med råd angående skriving av bachelor.

3.2 Oppsummering brukere

3.2.1 Prosjektledere

Prosjektledere og teamledere er brukerne av løsningen, og skal benytte seg av produktet gjennom å sende inn enten en CV til en ansatt eller flere CV-er til team. Visualisering av team kan deretter brukes i salgssammenheng for å visualisere hva ulike team i bedriften har å tilby. Visualiseringen kan også brukes til å utforske ulike teamsammensetning, bedriftens kompetanse i sin helhet og den spesifikkes ansattes kompetanse og erfaring. Dette kan igjen benyttes til å øke kompetansen til bedriften, team og ansatte hvor det trengst eller ses på som ønskelig. Det kan også effektivisere prosessen med å sette sammen team, og prosessen med å gi kunden et klart bilde på hvilken kompetanse og erfaring et team sitter

inne med.

Disse brukerne vil bli representert av produkteier og hans ansatte. Deres rolle under utviklingen vil være gjennom møter, brukertester og uformelle innspill for å sikre at funksjonaliteten vil bli utviklet på best mulig måte i forhold til deres behov.

3.2.2 Kunder av bedriften

Kunder til bedriften vil være indirekte brukere av løsningen, da de kan motta visualisering av team for å lettere kunne ta informerte valg om hvilke team som passer best i deres tilfelle. De vil ikke ha en rolle under utviklingen og derfor heller ikke representert. Det vil dog likevel være noe gruppen er oppmerksom på under utviklingen, med fokus på at visualiseringene skal ha logisk leselighet og design. Tietoevry har grafiske retningslinjer, og en visualisering som skal brukes ut mot kunder må naturligvis følge disse slik at produktet representerer bedriften.

3.3 Brukermiljøet

Brukeren vil i hovedsak forholde seg til produktet i en kontor-setting, der avgjørelser rundt sammensetning av et team skal tas. I en slik setting kan brukeren være i en presset sinnstilstand, der avgjørelsen som blir tatt vil ha konsekvenser for effektivitet og økonomi, samt kundens tilfredsstillelse. Det er også rimelig å anta at produktet vil brukes under oppsyn av andre ansatte eller oppdragsgivere.

Produktet vil bli bygd opp som en frittstående webapplikasjon, men brukeren vil også ha tilgang til Tietoevrys database med ansattes CV'er som et verktøy som kan brukes i kombinasjon med produktet.

3.4 Sammendrag av brukernes behov

Hovedbehovet til brukere, som er beskrevet i kapittel 3.2 Oppsummering av brukere, er å løse problemene beskrevet i kapittel 2.1 Problemsammendrag. Brukeren av systemet har behov for å effektivisere arbeidet med å sette sammen gode team, samt å visualiserer kompetanse og erfaring innad i team for kunder. Videre vil brukeren også ha behov for å visualisere en person, for å kunne få et godt overblikk over personens svakheter og styrker.

Oppdragsgiver ønsker at fokus hovedsakelig skal ligge på selve visualiseringen, samt muligheten for å eksportere denne til vektorgrafikk, slik at den kan brukes i salgssammenheng. Selve visualiseringen vil derfor ha høyeste prioritet. Brukerne av produktet vil i stor grad ha høy teknisk kompetanse, og behovet for eksepsjonelt gode brukergrensesnitt vil derfor prioriteres etter visualiseringen.

I dag har produkteier ingen måte å visualisere kunnskap på. Sammensetning av team skjer hovedsakelig ved at man leser prosatekst fra CVer. Det lages ikke visuelle fremstillinger av kompetansen hos en enkelt bruker. I enkelte tilfeller lages det visualiseringer av team manuelt. Dette skjer hovedsakelig i forbindelse med salg av team.

Dette produktet vil løse dette i stor grad ved å automatisere visualiseringen av CVer. Dette fører både til at det skal bli enklere å skaffe seg en oversikt over hvilken erfaring og kompetanse som finnes i team og enkeltpersoner. Det vil gjøre at man kan ta i bruk visualisering av kunnskap i forbindelse med å sette sammen team, samt at det vil gå med

mindre tid til å visualisere team i forbindelse med salg.

CVer kan inneholde personsensitiv informasjon. Dette må tas høyde for i produktet, slik at brukeren er beskyttet. Både informasjonen i brukersystemet, og informasjonen man henter fra CVer må beskyttes slik at den ikke er tilgjengelig for uvedkommende. Derfor skal ikke noe informasjon lagres før et brukersystem er på plass. Det skal være mulig, i henhold til de funksjonelle kravene i kapittel 5, å slette all informasjon som lagres om både personer representert ved en CV og brukere i brukersystemet. Visualiseringen skal basere seg på standardiserte CVer, og vil ikke vise noe mer informasjon enn disse CVene inneholder. De standardiserte CVene brukes allerede i dag til salg eksternt, og det vurderes derfor at det ikke er behov for noe ekstra filtrering av informasjon fra disse

3.5 Alternativer til vårt produkt

Det finnes produkter som visualiserer alt fra en enkelt persons kompetanse til å visualisere hele bedrifter. Det finnes mange muligheter for å lage en mer visuell CV og portefølje. Disse baserer seg på at man legger inn informasjon manuelt. Noen har støtte for at man kan koble seg til LinkedIn, for å automatisk hente ut relevant kompetanse og erfaring (Singh, u.d.). Disse tjenestene fokuserer på å visualisere en enkelt persons kompetanse og erfaringer gjerne til bruk i jobbsøking.

MuchSkills er en tjeneste som tilbyr å visualisere både enkeltpersoners harde og myke ferdigheter, samt en samlet visualisering av en hel bedrift (MuchSkills, u.d.). MuchSkills baserer seg på at hver enkelt ansatt i en bedrift har en bruker hvor de legger inn sin egen kompetanse i form av ferdigheter(skills) og hvilket nivå de selv rater seg. Ansvaret for å holde informasjonen oppdatert tilfaller den ansatte, og de får se sin kompetanse visualisert i sin profil. Bedriftens helhetlige spekter av kompetanse visualiseres ut i fra de samlede profilene på flere nivåer.

MuchSkills viser bedriftens samlet mengde av de ulike ferdighetene i kategorier. Videre kan man gå inn på en spesifikk kategori og sortere basert på avdelinger, erfaring og lokasjoner. Derfra kan man gå enda dypere inn på spesifikke ferdigheter, eller profiler. Gjennomgående i hele tjenestene kan man interaktivt trykke for å få opp navn, antall eller lister (MuchSkills, 2021). Denne tjenesten baserer seg mest på å visualisere en avdeling eller en bedrift helhetlig, og ikke nødvendigvis team som skal samarbeide.

Andre tjenester, som Vizual Workforce, baserer seg på at man som bedrift tar kontakt med dem for å få et spesialisert tilbud for bedriftens behov for visualisering av kompetanse (Visual Workforce, n.d.). Dette ligner delvis på løsningen foreslått i dette dokumentet, men for å få mer innsikt i deres tjenester trenger man en avtale. En slik avtale ble ikke inngått i forbindelse med utviklingen av dette dokumentet, og det mangler derfor utdypende informasjon om hvordan produktet fungerer.

4. Produktoversikt

4.1 Produktets rolle i brukermiljøet

I en presset situasjon er det mange hensyn produktet må ta stilling til. Det er derfor spesielt viktig at produktet oppfyller sin rolle som hjelpemiddel. Det er store mengder data som prosesseres, og i dette miljøet er det viktig at disse dataene blir fremstilt på en god måte. Ved å prosessere så mye data i en allerede stressende setting er det en fare for at brukeren blir kognitivt overanstrengt, noe som har negativ påvirkning på produktiviteten (Kirsh, 2000). I følge Farrington (2011) kan en menneskehjerne bare håndtere 3-4 nye biter av informasjon på en gang, og dersom produktet overstiger denne grensen vil risikoen for overanstrengelse øke.

Produktet skal være et verktøy som er godt integrert i arbeidshverdagen, som virker som et hjelpemiddel og ikke en hindring. Produktet skal fungere i sammenheng med bedriftens eksisterende systemer. Dette betyr at produktets funksjonalitet i første omgang skal virke utfyllende på den funksjonaliteten som allerede eksisterer.

4.2 Forutsetninger og avhengigheter

Brukeren må ha tilgang til en datamaskin som kan kjøre en moderne nettleser. Produktet kan også brukes på et nettbrett eller en telefon. Brukeren må også ha en admin-bruker i systemet for å få tilgang til sensitiv informasjon om ansatte. Under bruk må brukeren ha tilgang til internett.

For å utvikle prosjektet må utviklerne ha tilgang til datamateriale fra Tietoevry, i form av CV-databasen.

5. Produktets funksjonelle egenskaper

Nr	Funksjon	Detaljer
1	Logg inn	Logge inn i applikasjonen
2	Opprett person	Lager en ny person tilknyttet en CV
3	Last opp CV	Laster opp CV
4	Hent ut informasjon fra CV	Skjer automatisk når man laster opp CV
5	Hente navn	Finner navnet til personen fra CV
6	Hente mobilnummer	Hente ut mobilnummeret til personen fra CV
7	Hente e-post	Hente ut epost til person fra CV
8	Identifisere relevant erfaring i CV	Identifisere relevant informasjon i CVen
9	Rediger informasjon	Felt man kan endre på dersom data fra CV er feil.
10	Oppdater person	Oppdater informasjonen til en person ved å laste opp CVen på nytt. Forutsetter at personen er lagret
11	Visualiser en person	Visualiser kunnskapen til en spesifikk person
12	Visualiser kompetanse innenfor ulike kategorier	Visualiser hvilke forskjellige områder personen kompetanse på. Eksempel programmering, prosjektledelse og design.
13	Visualiser kompetanse innenfor ulike underkategorier	Visualiser hvilke forskjellige underkategorier personen har kompetanse på. Bryt ned kategoriene i mindre deler. Eksempel fullstack, backend og frontend, eller HTML, Vue og .NET.
14	Visualiser erfaring personen har innenfor ulike kategorier	Innenfor de ulike kategoriene har personer ulik erfaring. Beskriv denne erfaringen gjennom f.eks. pie chart eller lignende.
15	Visualiser erfaring personen har innenfor ulike underkategorier	Innenfor de ulike underkategoriene har personer ulik erfaring. Beskriv denne erfaringen i detalj gjennom f.eks. pie chart eller lignende.
16	Eksporter valgt diagram til vektorgrafikk	Eksporter valgt visualisering av kunnskapen til en enkeltperson som vektorgrafikk.

17	Eksporter alle diagram til vektorgrafikk	Eksporter alle genererte diagrammer til vektorgrafikk. Både hovedkategorier og underkategorier.
18	Last opp flere CVer og lag et team	Last opp mange CVer og sett alle sammen til et team (MVP)
19	Sett sammen team	List opp alle ansatte og sett sammen et team ut fra de ansatte. Med forutsetning at det finnes lagrede personer i systemet.
20	List opp ansatte/teammedlemmer	Vis alle ansatte. Forutsetning er at det finnes lagrede personer i systemet.
21	Visualiser et helt team basert på hovedkategorier	Visualiser kompetanse til et helt team basert på hovedkategorier. samt en oversikt over teammedlemmer. Et helt team er da 4-7 personer.
22	Visualiser et helt team basert på underkategorier	Visualiser kompetansen til et helt team basert på underkategorie
23	Velg om det skal visualiseres hovedkategorier eller underkategorier for team	Velg om man vil visualisere en grovoversikt med hovedkategorier eller vise underkategoriene for en spesifikk hovedkategori
24	Velg hovedkategori for å visualisere underkategorier	Velg en spesifikk hovedkategori for å visualisere underkategoriene
25	List opp teammedlemmer med navn og kontaktinformasjon	List opp alle teammedlemmer når man setter sammen et team slik at det er lett å holde oversikt over hvem som er på teamet
26	Slett person	Slett en person fra systemet. Forutsetter at lagring av personer er implementert
27	Logge ut	Logge ut
28	Slette bruker	Slette bruker med all tilhørende informasjon. Personer brukeren har opprettet i systemet slettes ikke når brukeren slettes.
29	Produktet skal støtte flere nettlesere	Produktet skal fungere i de vanligste nettleserne.

6. Ikke-funksjonelle egenskaper og andre krav

De ikke-funksjonelle egenskapene er strukturert etter FURPS. FURPS er et akronym for *functionality, usability, reliability, performance og supportability*. Produkteier har satt opp de forskjellige aspektene i en prioritert rekkefølge, slik at avgjørelsen ved eventuelle dilemmaer skal være i tråd med visjonen til produkteier. De fem punktene skal prioriteres i følgende rekkefølge:

1. *Functionality*
2. *Supportability*
3. *Usability*
4. *Reliability*
5. *Performance*

6.1 Functionality

For god funksjonalitet er det viktig med god sikkerhet. Produktet skal utvikles som en web applikasjon. Produktet skal også prosessere og lagre informasjon om ansatte og deres CV. Derfor er det viktig med god sikkerhet i applikasjonen. For å oppnå god sikkerhet, følges retningslinjene Open Web Application Security Project (OWASP) for .NET og ASP.NET.

Når man lager web-applikasjoner med ASP.NET, må det tas noen hensyn med tanke på sikkerhet. Et av de vanligste grepene som må gjøre for å sikre seg mot angrep er å bruke parametriserte SQL-spørringer (DotNet Security - OWASP Cheat Sheet Series, u.d.). En bør unngå bruk av dynamiske SQL-spørringer. Man bør også sjekke all input fra bruker ved å definere tillatte verdier. Når man skal koble på databasen skal prinsippet med "minste privilegium brukes". Det betyr at koblingen til databasen kun skal ha tilgang til det den trenger (DotNet Security - OWASP Cheat Sheet Series, u.d.). Det skal brukes Entity rammeverk for tilkobling til databasen. Dette gjøres for å hindre SQL injeksjoner. For å beskytte data, skal alle sensitive data krypteres (DotNet Security - OWASP Cheat Sheet Series, u.d.).

Kryptering skal aldri skrives på egenhånd, da dette kan generere sikkerhetshull (DotNet Security - OWASP Cheat Sheet Series, u.d.). Azure vil benyttes for lagring av data. Azure tilbyr funksjonalitet som nøkler og managed identities, som ivaretar mange utfordringer knyttet til sikkerhet ved brukere og brukertilgang (Baldwin, 2022). ASP.NET Core Data Protection sammen med Azure Key Vault og Azure Storage brukes for å lagre nøkler på en sikker måte (Rick-Anderson, 2020), (Neria, Chong and Hemat, 2022). Det er også viktig å bruke gode algoritmer for å hashe, samt å alltid salte passord (DotNet Security - OWASP Cheat Sheet Series, u.d.). Det er også viktig at konfigurasjonsfilen ikke inneholder kode som ikke brukes, og at sensitiv informasjon ikke er tilgjengelig gjennom filen (DotNet Security - OWASP Cheat Sheet Series, u.d.). For å oppnå god sikkerhet med en

web-applikasjon skal HTTPS benyttes (DotNet Security - OWASP Cheat Sheet Series, u.d.).

For brukersystemet skal gjeldene krav spesifisert i GDPR innfris (*Lov om behandling av personopplysninger (personopplysningsloven) - EUROPAPARLAMENTS- OG RÅDSFORORDNING (EU) 2016/679 av 27. april 2016 om vern av fysiske personer i... - Lovdata*, u.d.). Videre er funksjonalitet spesifisert i kapittel 5 Funksjonelle krav.

6.2 UX user experience

Tietoevry har lagt frem et krav om at produktet skal utformes etter Tietoevry sin eksisterende designmal. Som nevnt i kapittel 4.1 Produktets rolle i brukermiljøet, er det ekstra hensyn som må tas på grunnlag av brukermiljøet, der produktet skal være utformet slik kognitiv *overload* unngås. For å oppnå dette skal produktet presentere dataene på en måte som er overkommelig for brukeren å prosessere. For å oppnå dette kan man benytte seg av prinsippene innenfor calm technology. Disse prinsippene går ut på at teknologi ikke skal oppta mer av brukerens oppmerksomhet enn nødvendig, for å bidra til et hyggeligere brukermiljø (*Calm Technology*, u.d.). Weinberg (2019) påpeker at interaksjonen med et user interface kan ha stor innvirkning på en menneskets sinnstilstand, og ved å bevisst utvikle et produkt med hensyn til brukermiljøet kan man oppnå gode resultater.

Utover dette skal produktet være responsivt, slik at det er enkelt å forstå. Knapper og andre felter som tar imot input fra brukeren skal, ved å være responsive, gi tilbakemelding til brukeren når de blir trykket slik at brukeren forstår at input har blitt registrert.

For å sikre at produktet er universelt utformet skal det oppfylle minstekravet for Web Content Accessibility Guidelines. Produktet skal være universelt utformet for å sørge for at flest mulig har tilgang til å bruke det. Dette innebærer at 35 spesifikke krav må innfris. Disse kravene listes opp på tilsynet for universell utforming av IKT sine nettsider under WCAG 2.0 Standarden (WCAG 2.0-standarden, u.d.).

6.3 Reliability

Produktet tar sikte på å ikke ha mer enn 5% av driftstid nede. . Dette vil bli vanskelig å vite om man kommer i mål med, da produktutviklingsfasen er over en såpass kort periode at et produkt med tilstrekkelig implementert funksjonalitet ikke vil være på plass tidlig nok til at kravet kan måles. Likevel settes dette som et overordnet krav. Det stiller krav til at det skrives kode som er leselig, slik at det er lett å oppdage logiske feil ved koden.

6.4 Performance

Produktet skal ta sikte på å ha en rask reaksjonstid. Om produktet i enkelte sammenhenger bruker lang tid på å utføre en handling utført av brukeren, skal det være tydelig for brukeren at systemet jobber med noe. For prosesser som tar lang tid skal det gis tilbakemelding på skjermen. Systemet skal ta sikte på å være skalerbart. Systemet skal

være skalerbart for å sørge for at det både tåler en økning i trafikk, samt at det kan skaleres ned for å redusere kostnader ved lite bruk (*Definition of Scalability - Gartner Information Technology Glossary*, u.d.).

6.5 Supportability

Produktet som utvikles skal utvikles med tilstrekkelig tester slik at man tidlig kan oppdage feil i koden. Produktet skal også utvikles slik at det er testbart. Produktet skal utvikles som en web-applikasjon som skal fungere i de mest brukte nettleserne. I 2021 hadde Google Chrome en markedsandel på 63%, Safari hadde 20% og Mozilla firefox og Microsoft edge hadde henholdsvis 4% hver (*Browser Market Share Worldwide*, u.d.). Det er dermed naturlig å prioritere Chrome og Safari under utviklingen av produktet. Produktet skal utvikles med norsk som språk. Det tas i første omgang ikke sikte på å legge til rette for å implementere flere språk.

Node.js er et populært verktøy for utvikling av webapplikasjoner, og et verktøy som vil bli brukt også i dette prosjektet. Det rimelig å anta at man må ta i bruk tredjeparts biblioteker for å komme i mål, fordi node.js har relativt lite funksjonalitet i sitt bibliotek. Dette fører med seg litt hodebry i forhold til vedlikehold (Says, 2020). Dersom *dependencies* blir utdaterte kan applikasjonen få problemer med sikkerhetshull, eller metoder som ikke lengre fungerer. Heldigvis har Node.js mange gode verktøy for å løse dette problemet, som kan benyttes under dette prosjektet (*The Node.js Application Maintainer Guide*, 2021).

7. Referanser

Baldwin, M. (2022) *Azure Key Vault managed storage account - PowerShell version*, *Microsoft docs*. Tilgjengelig på: https://docs.microsoft.com/en-us/azure/key-vault/secrets/overview-storage-keys-powershell (Hentet: 3 Februar 2022).

Browser Market Share Worldwide (u.d.) *StatCounter Global Stats*. Tilgjengelig på: https://gs.statcounter.com/browser-market-share (Hentet: 4 Februar 2022).

Calm Technology (u.d.). Tilgjengelig på: https://calmtech.com/ (Hentet: 4 Februar 2022).

Definition of Scalability - Gartner Information Technology Glossary (u.d.) *Gartner*. Tilgjengelig på: https://www.gartner.com/en/information-technology/glossary/scalability (Hentet: 4 February 2022).

DotNet Security - OWASP Cheat Sheet Series (u.d.). Tilgjengelig på: https://cheatsheetseries.owasp.org/cheatsheets/DotNet_Security_Cheat_Sheet.html (Hentet: 3 Februar 2022).

Farrington, J. (2011) 'Seven plus or minus two', *Performance Improvement Quarterly*, 23(4), pp. 113–116. doi:10.1002/piq.20099.

Kirsh, D. (2000) 'A Few Thoughts on Cognitive Overload', *Intellectica*, 1(30), pp. 19–51.

Lov om behandling av personopplysninger (personopplysningsloven) - EUROPAPARLAMENTS- OG RÅDSFORORDNING (EU) 2016/679 av 27. april 2016 om vern av fysiske personer i... - Lovdata (u.d.). Tilgjengelig på: https://lovdata.no/dokument/NL/lov/2018-06-15-38/gdpr#gdpr (Hentet: 11 Mars 2022).

MuchSkills [MuchSkills]. (27. sep 2021). *MuchSkills 3.0 : The future of skill mapping is finally here* [Video]. YouTube. https://www.youtube.com/watch?v=pLdXsNx3o1Y

Muchskills.com (2022) *MuchSkills : Skills Management Platform.* Tilgjengelig på: https://www.muchskills.com/ (Hentet 3 Februar 2022).

Neria, B., Chong, K. and Hemat, A. (2022) *Managed identities for Azure resources*. Tilgjengelig på: https://docs.microsoft.com/en-us/azure/active-directory/managed-identities-azure-resources/overview (Hentet: 3 Februar 2022).

Rick-Anderson (2020) *ASP.NET Core Data Protection*. Tilgjegelig på: https://docs.microsoft.com/en-us/aspnet/core/security/data-protection/introduction (Hentet: 3 Februar 2022).

Says, A.R.G. (2020) *How to keep your JavaScript libraries up to date*, *LogRocket Blog*. Tilgjengelig på: https://blog.logrocket.com/how-to-keep-javascript-libraries-up-to-date/ (Hentet: 4 Februar 2022).

Singh, R. (u.d.) *5 Great Tools to Turn Your Resume Into a Visual Masterpiece.* Tilgjengelig på: https://www.recruiter.com/i/5-great-tools-to-turn-your-resume-into-a-visual-masterpiece/ (Hentet 3 Februar 2022)

The Node.js Application Maintainer Guide (2021) *The NodeSource Blog - Node.js Tutorials, Guides, and Updates*. Tilgjengelig på: http://nodesource.com/blog/the-Node.js-application-maintainer-guide/ (Hentet: 4 Februar 2022).

Tietoevry.com (u.d.) *Tietoevry skaper meningsfull teknologi som bidrar til å gjøre verden bedre*. Tilgjengelig på: https://www.tietoevry.com/no/om-oss/om-tietoevry/ (Hentet: 28 Januar 2022).

Visual Workforce (u.d.) *Product — Visual Workforce.* Tilgjengelig på:

https://www.visualworkforce.com/product (Hentet 3 Februar 2022).

WCAG 2.0-standarden | Tilsynet for universell utforming av ikt (u.d.). Tilgjengelig på: https://www.uutilsynet.no/wcag-standarden/wcag-20-standarden/86 (Hentet: 3 Februar 2022).

Weinberg, M. (2019) *Designing for user environments*, *Medium*. Tilgjengelig på: https://uxdesign.cc/designing-for-user-environments-663b1f8e782e (Hentet: 3 Februar 2022).

Weise, M. (2016) 'We Need a Better Way to Visualize People's Skills', *Harvard Business Review*, 20 September. Tilgjengelig på: https://hbr.org/2016/09/we-need-a-better-way-to-visualize-peoples-skills (Hentet: 3 Februar 2022).

Vedlegg 3

59

Visualisering av kompetanse
Kravdokumentasjon

Versjon 1.0

Revisjonshistorie

Dato	Versjon	Beskrivelse	Forfatter
04.02.2022	0.1	Første utkast	Jenny F. B.
06.02.2022	0.2	User stories, wireframes	Jenny F. B.
11.02.2022	0.3	Skriver tekst, legger til figma prototype.	Linda. K. L., Jenny F. B.
17.02.2022	0.4	Datamodellering og finskriving	Jenny F. B.
18.02.2022	0.5	Oppdaterer userstories, wireframes.	Jenny F. B.
11.03.2022	0.6	Ferdigstiller protoryper og wireframes.	Jenny F. B.

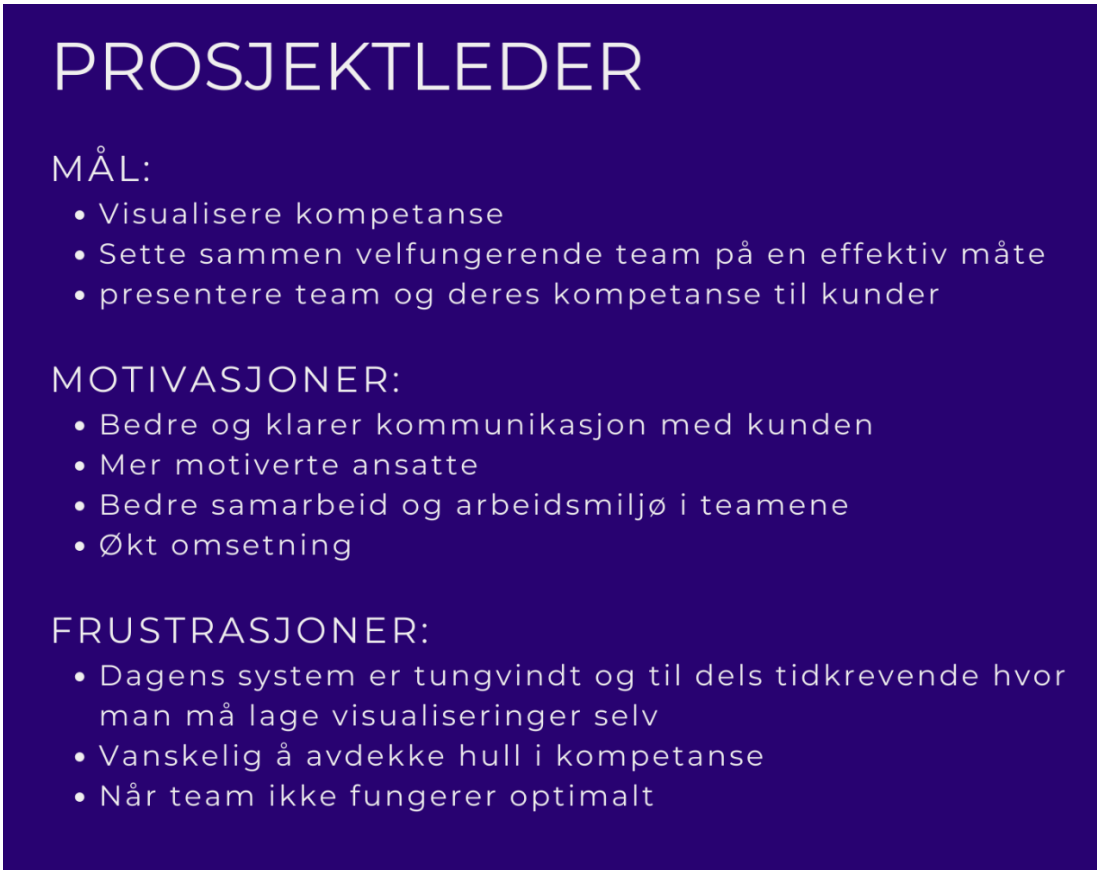
Innholdsfortegnelse

1. Introduksjon

Dette kravsdokumentet er blitt utarbeidet i sammenheng med bacheloroppgaven Visualisering av Kompetane våren 2022. Hensikten med dette dokumentet er å beskrive de funksjonelle kravene til løsningen. Kravene vil bli lagt frem gjennom userstories, backlogg, wireframes og prototyper i figma.

2. Personas

For å legge grunnlaget for user stories og de funksjonelle kravene ble det utarbeidet persona av brukeren prosjektleder (Figur 1). Dette er en prosjektleder innen it-bransjen, med hovedvekt på it-tjenester, og er brukeren som vil stå i hovedfokus under utviklingen. I figur 1, vises prosjektlederens mål, motivasjoner og frustrasjoner. Hovedmålet til prosjektlederen er å visualisere kompetansen til et team. Utover dette vil det også være et ønske og mål for prosjektlederen å sette sammen team på en effektiv måte og å kunne sammenligne ulike teamsammensetninger som resulterer i gode og velfungerende team.



Figur 1: Persona av brukeren prosjektleder

3. User Stories

Hovedmålet til prosjektet, og oppgavens "epic" er som følger; "Som prosjektleder vil jeg visualisere kompetanse slik at jeg kan holde oversikt over ansattes kompetanse og erfaring som igjen kan brukes til å kommunisere hva ansatte og team kan til kunder, samt til å sette sammen nye team".

Userstoriesene er brutt ned fra epic-en og beskrives i tabellen under. I user-storyene blir prosjektlederens ønsker og mål beskrevet nærmere med akseptansekriterier.

Som prosjektleder:

Ønsker jeg...	**slik at...**	**Akseptansekriterier**
å laste opp en CV	informasjonen den inneholder blir visualisert	- Mulig å laste opp en CV - Mulig å lagre og behandle informasjonen i CV-en
å se en ansatt sin kompetanse og erfaring visualisert	jeg har oversikt over deres kompetanse og erfaring	- Hull i kompetansen kan avdekkes - Kompetansen visualiseres i kategorier og underkategorier - Erfaringen innenfor ulike kategorier visualiseres
å se et team sin kompetanse og erfaring visualisert	jeg har oversikt over teamets kompetanse og kan lettere formidle dette	- Flere CV-er kan lastes opp samtidig. (2-7 stk) - Evt. hull i kompetansen til teamet visualiseres - Kompetansen visualiseres i kategorier og underkategorier - Erfaringen til teamet innenfor de ulike kategoriene visualiseres.
å kunne laste ned visualiseringer	jeg enkelt kan lagre og bruke dem senere	- Eksportering til vektrografikk - Mulig å laste ned all visualisering tilhørende ansatt og

		team - Mulig å laste ned spesifikke diagrammer og visualiseringer
å lagre informasjon om ansatte i systemet	at de lettere kan settes sammen i ulike team	- Innlogget som prosjektleder kan man lagre, oppdatere og slette ansatte i systemet. - Oppretting og oppdatering av ansatte skjer gjennom parsing av CV-er
å kunne sette sammen team	jeg enkelt kan sette sammen nye team og evt. sammenligne ulike sammensetninger	- mulig å sette sammen team fra ansatte lagret i systemet.

4. Backlogg

Ut fra User storisene og de funksjonelle kravene i visjonsdokumentet ble produkt backlogg utarbeidet. Resultatet av produkt backloggen ble lagt inn i Azure devops og deretter strukturert i epics, features og backlog items. Hvert backlogg item har en estimert tid og verdi, og ut fra dette fått dens prioritering.

På figur 2 sees produkt backlogg med overordnet epics med tilhørende features. Den første itemen kan sees bort ifra, da denne vil brukes til å administrere annet arbeid utført utenom utviklingen av produktet. Featursene legges inn med tilhørende user story og akseptansekriterier.

Order	Work Item Type	Title
1	Epic	Fremdriftsplan
2	Epic	Visualisering
	Feature	Visualisering av team
	Feature	Visualisering av ansatt
	Feature	Presentasjon for bruker
	Feature	Bygning av visuelle modeller fra datamodellene
3	Epic	Støttefunksjoner
	Feature	Layout og sammensetning
	Feature	Eksportering
	Feature	Brukerhåndtering
4	Epic	CV-prosessering
	Feature	redigere informasjon fra generert CV
	Feature	Algoritme som ranker og tygger på innholdet
	Feature	Parsing av CV
	Feature	Datamodeller

Figur 2: Product backlogg på overordnet nivå i Azure devops

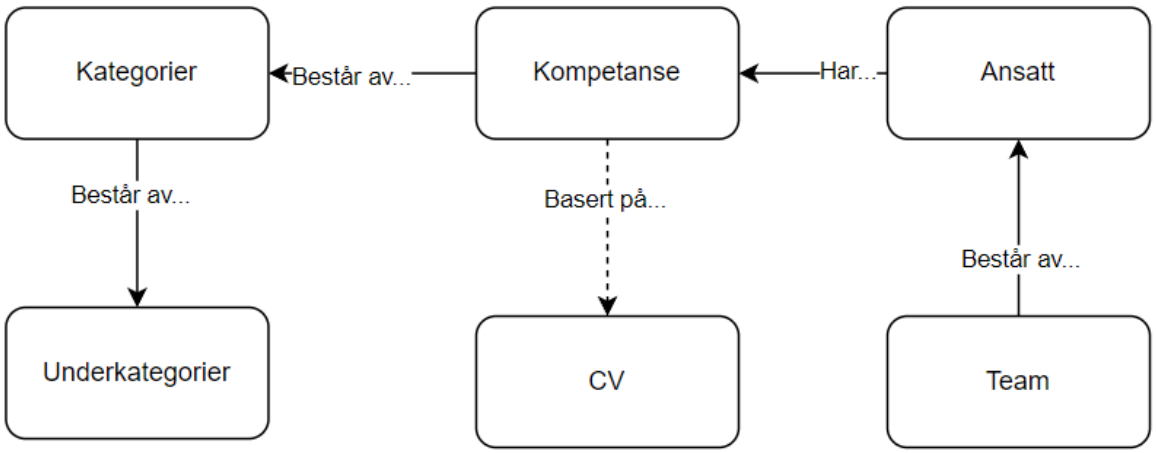
Hver feature har tilhørende backlogg-items. På Figur 3 vises backlog-itemsa inne i Azure devops i prioritert rekkefølge. De er merket med *effort* og *buisness value* som henholdsvis er esitmert tid og verdi.

Order	Title	Effort	Iteration Path	Business Value	Priority
1	Layout	40	Visualisering av Kompetanse\It...	60	1
2	Laste opp en CV	8	Visualisering av Kompetanse\It...	100	1
3	Laste opp flere CV-er	5	Visualisering av Kompetanse\It...	40	2
4	Visualisere kompetanse til en ansatt på et overordnet nivå	100	Visualisering av Kompetanse\It...	100	1
5	Eksportere all visualisering tilhørende en ansatt	40	Visualisering av Kompetanse\It...	60	2
6	Eksportere spesifikke diagrammer og visualiseringer tilhøren...	3	Visualisering av Kompetanse\It...	13	2
7	Visualisere kompetanse til et team på overordnet nivå	40	Visualisering av Kompetanse\It...	40	2
8	Eksporter all visualisering tilhørende et team	8	Visualisering av Kompetanse\It...	60	2
9	Eksportere spesifikke diagrammer og visualiseringer tilhøren...	3	Visualisering av Kompetanse\It...	13	2
10	Visualisere kompetanse til ansatt på et detaljert nivå	60	Visualisering av Kompetanse\It...	40	2
11	Visualiser kompetanse til et team på detaljert nivå	40	Visualisering av Kompetanse\It...	20	2
12	Visualiser erfaring til ansatt på et overordnet nivå	60	Visualisering av Kompetanse\It...	40	2
13	Visualiser erfaring til team på overordnet nivå	20	Visualisering av Kompetanse\It...	20	2
14	Se kontakt info til team-medlemmer	8	Visualisering av Kompetanse\It...	5	3
15	Visualiser erfaring til en ansatt på et detaljert nivå	40	Visualisering av Kompetanse\It...	20	2
16	Visualiser erfaring til et team på detaljert nivå	20	Visualisering av Kompetanse\It...	13	2
17	Bruker i systemet	40	Visualisering av Kompetanse\It...	60	3
18	Navigering mellom team og teammedlemmer	5	Visualisering av Kompetanse\It...	20	3
19	Navigere mellom ulike visualiseringer	8	Visualisering av Kompetanse\It...	13	3
20	Lagre ansatt	20	Visualisering av Kompetanse\It...	40	3

Figur 3: Produkt backlogg på detaljert nivå i Azure Devops.

5. Datamodellering

I denne oppgaven vil det arbeides med forskjellige entiteter og informasjon knyttet til dem. I figur 4 blir forholdene mellom ansatt, team, kompetanse, kategorier, underkategorier og CV beskrevet. Her ser vi at *Team* består av *Ansatte*, *Ansatte* har *Kompetanse* som igjen er basert på *CV*-er. Koblingen mellom *Kompetanse* og *CV* er stiplet siden de vil være tilnærmet ekvivalente i systemet. *Team* har også indirekte kompetanse. Den baseres på de ansatte i teamet sin *Kompetanse*. *Kompetanse* består igjen av *Kategorier* og *Underkategorier*. Siden prosjektet har en agil prosess vil entitetene *Kategorier* og *Underkategorier* utvikle seg iterativ gjennom prosjekperioden basert på hvordan visualiseringen av kompetansen utvikler seg.



Figur 4: En forenklet datamodell.

6. Prototyper

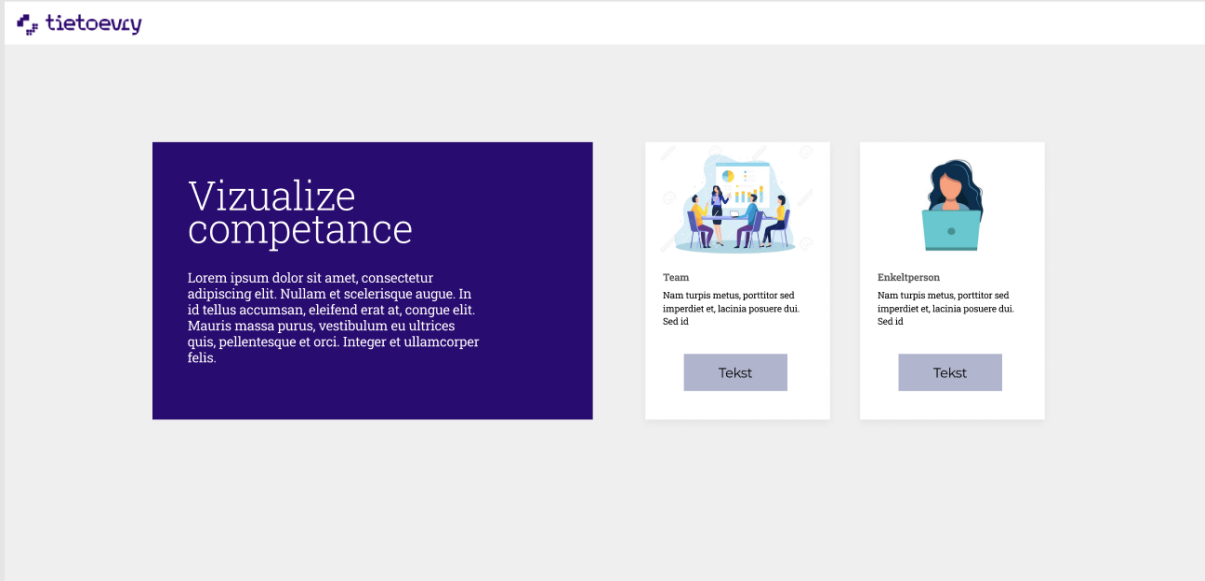
Prototyper av produktet gir et innblikk i hvordan det endelige produktet kommer til å ende opp. I dette prosjektet er det blitt utarbeidet flere wireframes og prototyper. Som følge av prosjektet iterative natur, vil disse bli utviklet underveis -- progresjonen til prosjektet. For hver Sprint skal det utvikles wireframes som dekker funksjonalitet i den kommende Sprinten, der disse kan utvikles underveis ettersom man får kjørt brukertester.

1.1 Figma-prototype

I lenken under ligger prototypen til prosjektet. Prototypen følger de grafisk retningslinjene til tieotevry og de krav som er satt til brukervennelighet. Man kan interaktivt trykke seg gjennom prototypen ved å trykke på play knappen oppe i høyre hjørne. Siden prosjektet følger en agil prosess, vil kravene og prototypen endre seg iterativ gjennom prosessen. Hvordan prototypen har utviklet seg vil vises i ulike kategorier av figma prosjektet.

Link til figma-prototype

https://www.figma.com/file/gcy7HGf62zoLcPzgXEpp8u/Visualize-competance?node-id=0%3A1



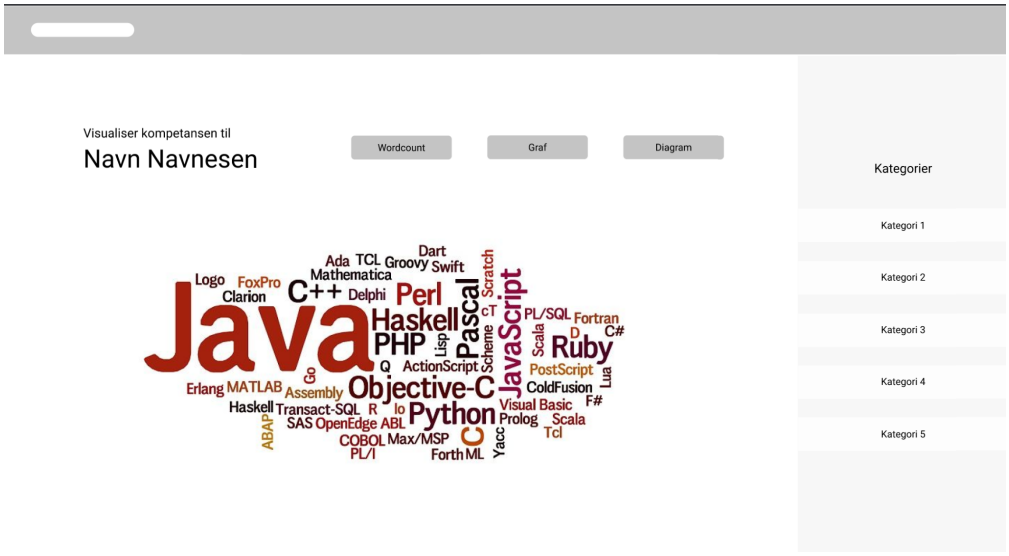
Figur 5: Skjermbilde av første utkast av en prototype til prosjektet.

1.2 Wireframes

Det ble utviklet flere forskjellige wireframes med ulike formål, men felles mål om å skape en god basis og tydeligjøre kravene til produktet.

1.2.1 Brukertesting

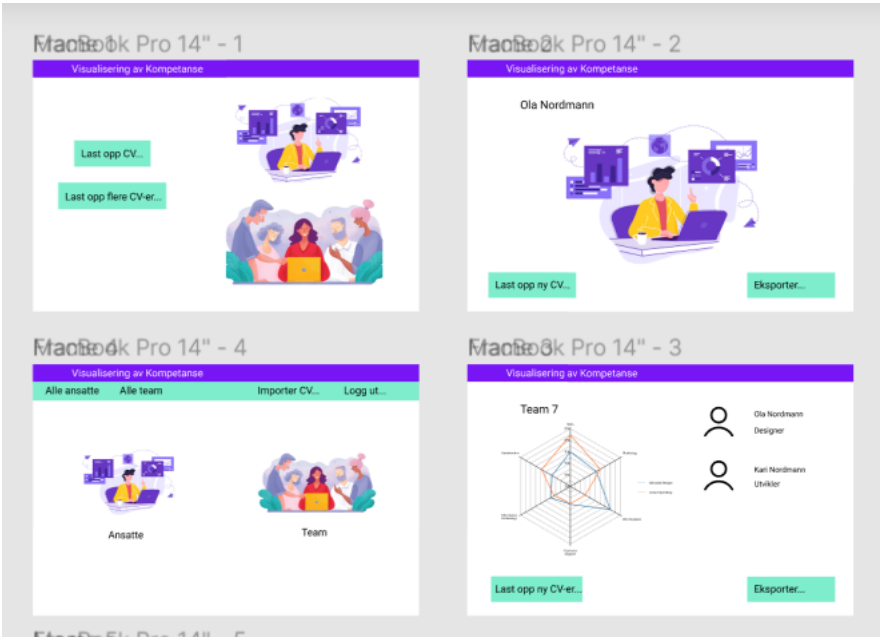
For å kunne holde brukertester hvor funksjonaliteten står i fokus ble det utarbeidet en wireframe med minst mulig farger og andre distraksjoner. På figur 6 sees et skjermbilde fra siden med en foreslått visualisering av kompetanse til en ansatt fra denne wireframsen.



Figur 6: Skjermbilde av wireframes tiltenkt brukertest i prosjektet.

Link til brukertest wireframes:
https://www.figma.com/proto/ZLro0YMY1mgu8y760kmWKQ/Wireframes-brukertesting?node-id=2%3A2&scaling=min-zoom&page-id=0%3A1&starting-point-node-id=2%3A2

Underveis i prosessen laget alle medlemmene i gruppen ulike forslag til hvordan de ser for seg applikasjonens flyt og utseende. På figur 7 ser du et av disse utkastene. Disse ble presentert og diskuter av gruppen for å samles rundt en felles bilde for basisen til webapplikasjonen.

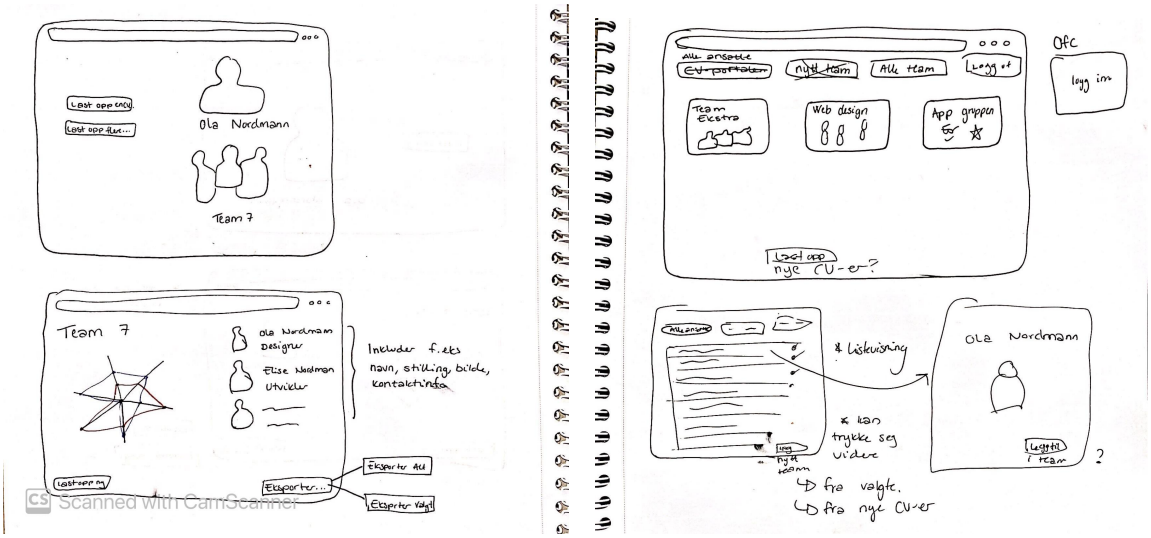


Figur 7: Skjermbilde av første ide av hvordan applikasjonen skal se ut.

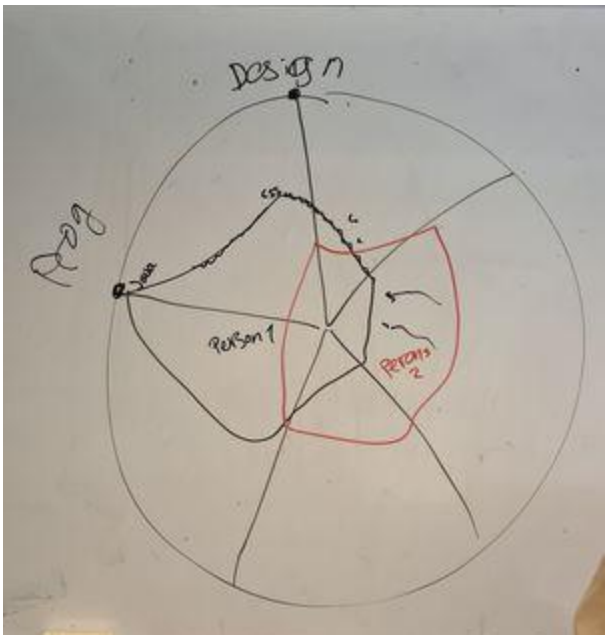
Link til wireframe-utkastet på figur 7:
https://www.figma.com/file/o60oL3VIUxP8ESHeRK33AO/Visualisering-av-Kompetanse?node-id=0%3A1

1.2.2 Low-fidelity

På figur 8, 9 og 10 sees de aller første og "lavest"-fidelity wireframene. Disse er da tegnet for hånd på papir og tavle.



Figur 8 og 9: Low fidelity wireframses.



Figur 10: Visualisering av team, første utkast.

59

Visualisering av Kompetanse
Systemdokumentasjon

Versjon 1.0

Revisjonshistorie

Dato	Versjon	Beskrivelse	Forfatter
29/04/2022	0.1	Første utkast	Jenny F. Blindheimsvik
13/05/2022	0.2	Revideres etter produktutviklingens slutt.	Jenny F. Blindheimsvik
18/05/2022	0.3	Ferdigstilles før innlevering.	Jenny F. Blindheimsvik
19/05/2022	1.0	Siste finpuss før innlevering.	Jenny F. Blindheimsvik, Nora E. Jansrud, Linda K. Larsen.

Innholdsfortegnelse

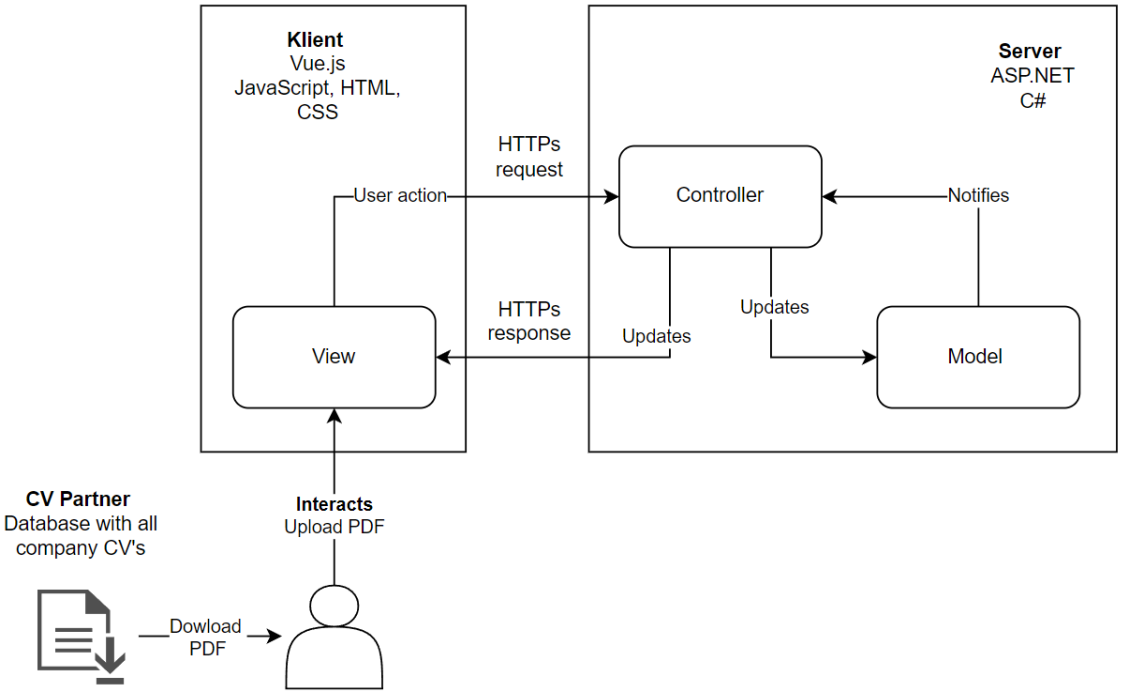
1. Introduksjon

Dette dokumentet er skrevet i forbindelse med gjennomføring av Bacheloroppgave IDATT2900 ved studiet Dataingeniør, NTNU, våren 2022. Hensikten med dokumentet er å beskrive systemet som er utviklet som en løsning på oppgaven *Visualisering av Kompetanse* stilt av Tietoevry Norway AS, herfra referert til som Tietoevry. Dokumentet inneholder prosjektets arkitektur, prosjektstruktur, klassediagram for serversiden, server-tjenester, sikkerhet, testing samt hvordan det installeres og kjøres.

2. Arkitektur

2.1 Overordnet arkitektur

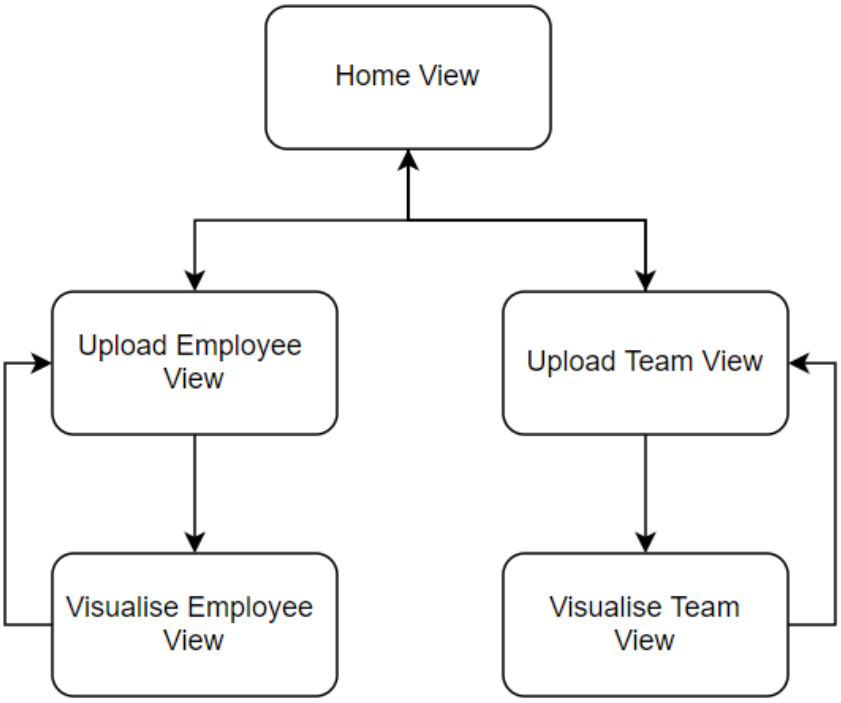
Systemet er brukt MVC (Model View Controller) design samt klient-tjener arkitektur til å bygge systemet hvor kommunikasjonen mellom klient og server skjer gjennom HTTPS kall og svar. Den overordnede arkitekturen kan sees på figur nr. 1. Datagrunnlaget til systemet er CVer som brukeren selv laster opp. CV-ene ligger tilgjengelig i tjenesten CV-Partner, som prosjektledere ansatt hos oppdragsgiver har tilgang til.



Figur nr. 1: *Systemets overordnede arkitektur.*

2.2 Klient-arkitektur

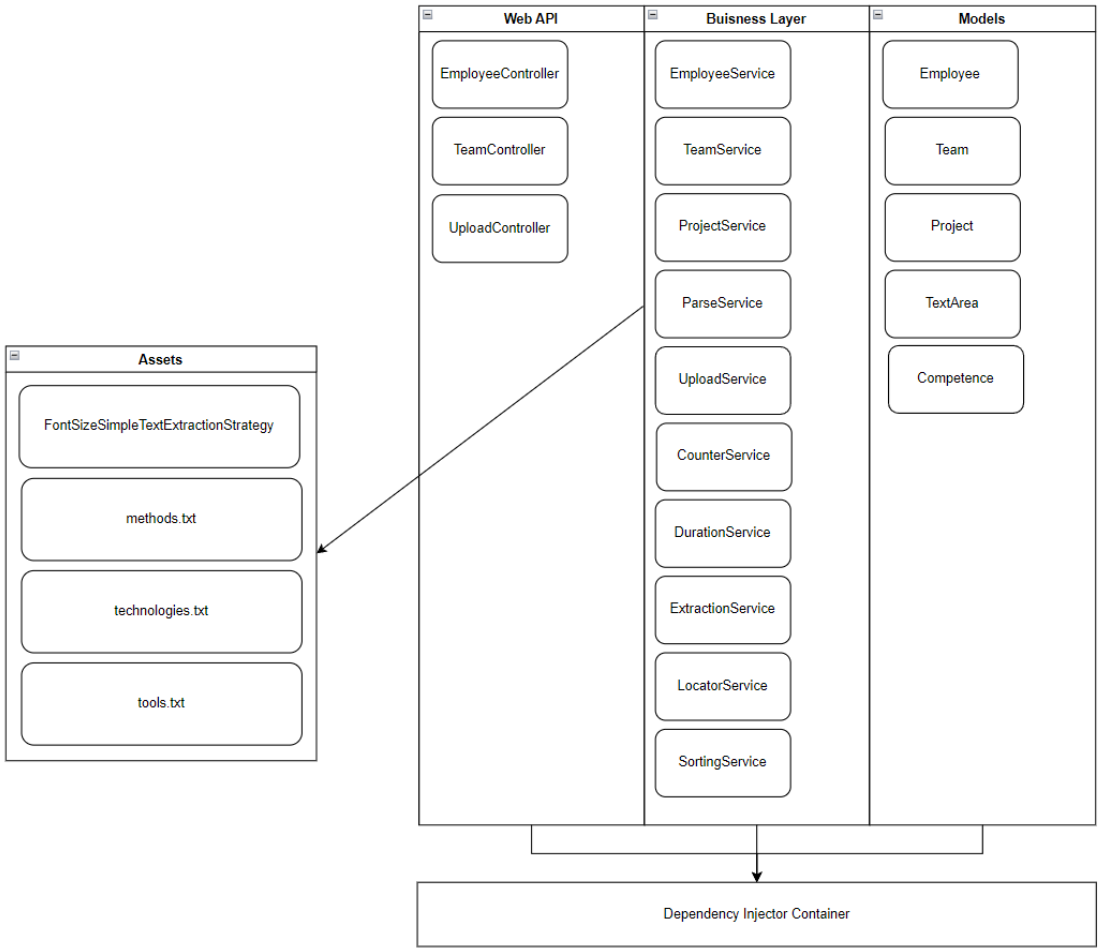
Klienten starter hos *Home view*. Der kan man velge om man ønsker å visualisere en enkelt ansatt, eller et team. Deretter havner man på opplastingssiden. På opplastingssiden hos en enkeltperson er det kun mulig å laste opp en pdf, mens for team kan man laste opp mange. Når man er ferdig med å laste opp CVer trykker man seg videre, og kommer frem til visualiseringene. Man kan gå tilbake og visualisere nye team eller ansatte.



Figur nr. 2: *Arkitektur for klienten*

2.3 Serverarkitektur

Backend er bygget etter REST prinsippene. Strukturen er lagdelt med *Controller, Service* og *Model*. *Controlleren* tar imot alle forespørsler fra klienten og sender dem videre til aktuell service-klasse. Service-klassene håndterer selve *business* logikken som omfatter blant annet opplastning og uthenting av informasjon fra CVene. Model-laget håndterer selve objektene. *Assets* inneholder en tilpasset klasse for uthenting av tekst fra CVene basert på skriftstørrelsen. Videre inneholder den standardiserte lister med oversikt over kategoriene metoder, teknologier og verktøy. *Dependecy injector container* representerer alle abstraksjonene i systemet som benyttes til å injisere avhengigheter inn av rammeverket. Serverens arkitektur kan sees på figur nr. 3.

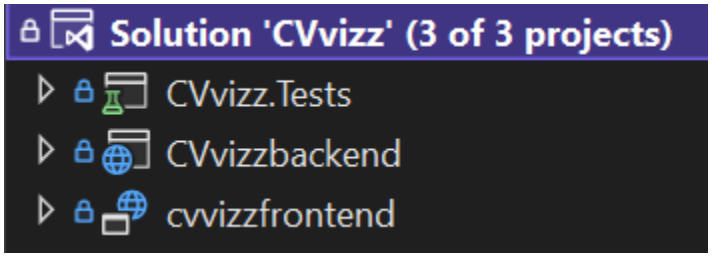


Figur nr 3: *Serverens arkitektur.*

3. Prosjektstruktur

3.1 Overorndet prosjektstruktur

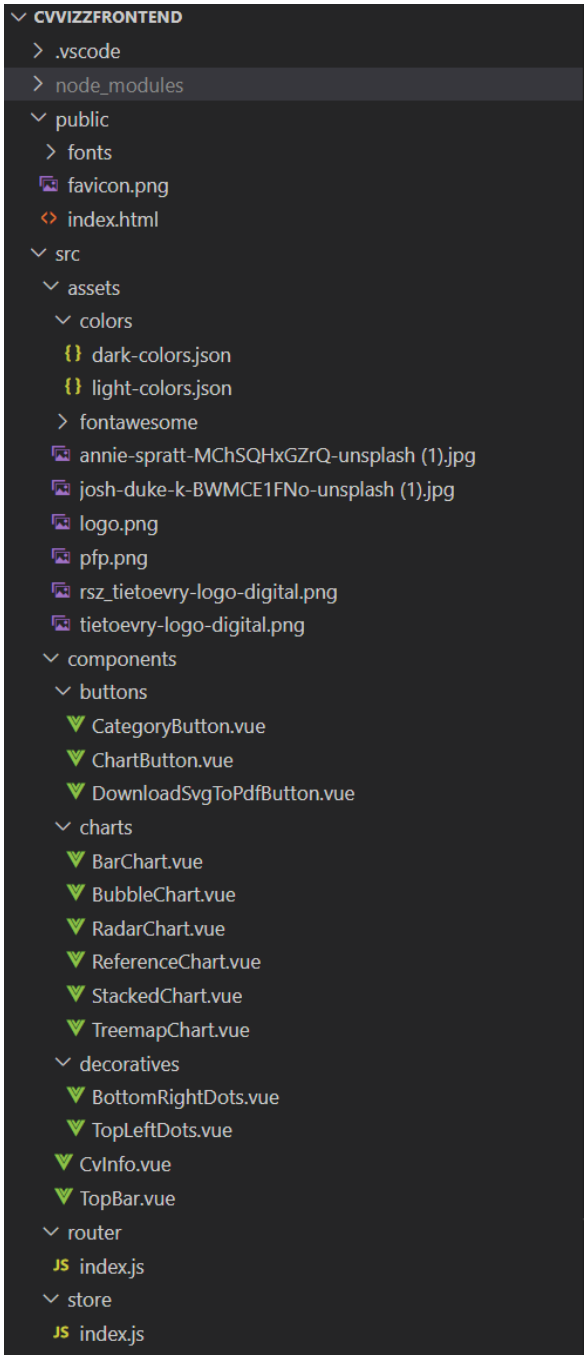
Den overordnede prosjektstrukturen til systemet består av tre prosjekter. Serveren, *CVvizzbackend*, klienten, *CVvizzfrontend*, og et for testing, *CVvizz.Tests*.

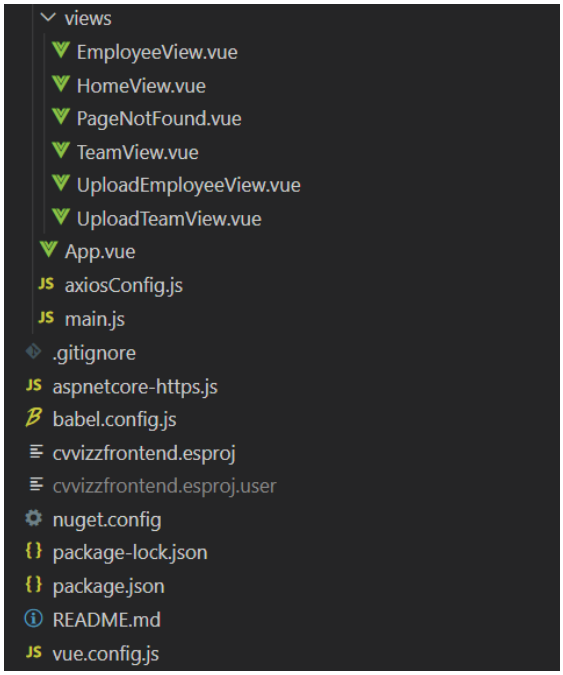


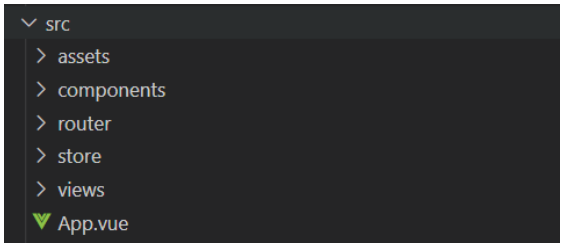
Figur nr. 4: *Overordnet prosjektstruktur.*

3.2 Prosjektstruktur til Klient

Klienten i systemet er delt inn *assets, components, router, store* og *views. Components* inneholder alle komponentene. *Views* inneholder alle sidene. *Router* til å rute mellom sidene, og *Store* til mellomlagring. Til slutt er mappen *assets* for andre ressurser.



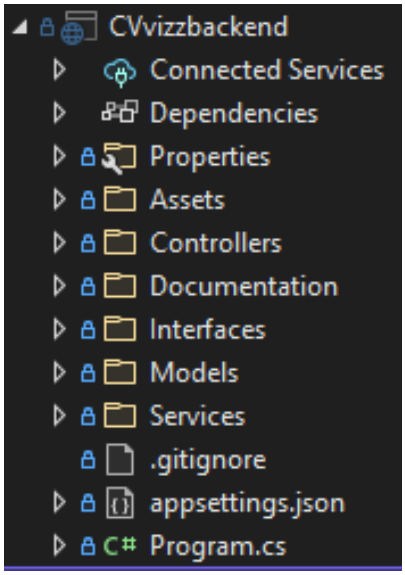




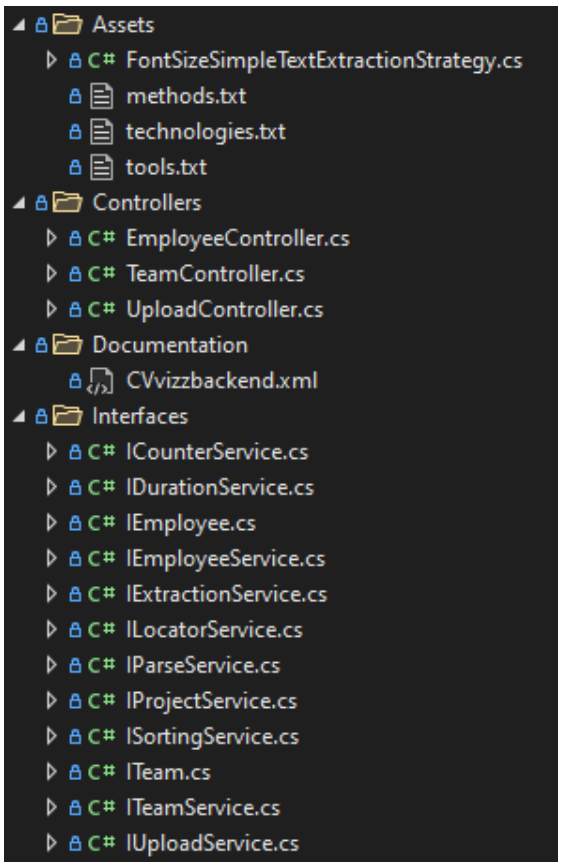
Figur nr. 5, 6 og 7: *Klientens prosjektstruktur.*

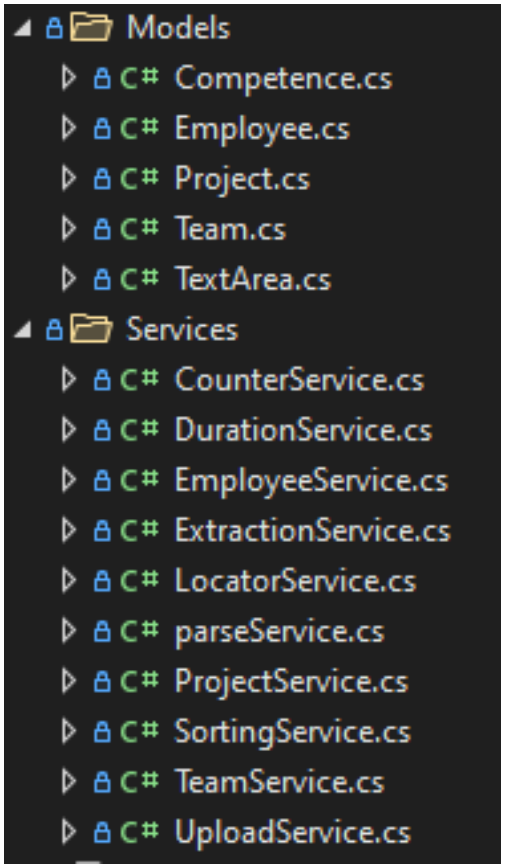
3.3 Prosjektstruktur til Server

Prosjektstrukturen til serveren er delt inn i *Assets, Controllers, Interfaces, Models* og *Services. Controller* inneholder alle kontroller-klassene som lytter på de ulike endepunktene. *Services* inneholder alle service-klassene som håndterer business logikken. *Models* inneholder aller modellene. Videre har *Interfaces* alle grensesnittene. Til slutt har mappen *Assets* alle andre ressurser programmet behøver.



Figur nr. 8: *Serverens overordnet prosjektstruktur.*

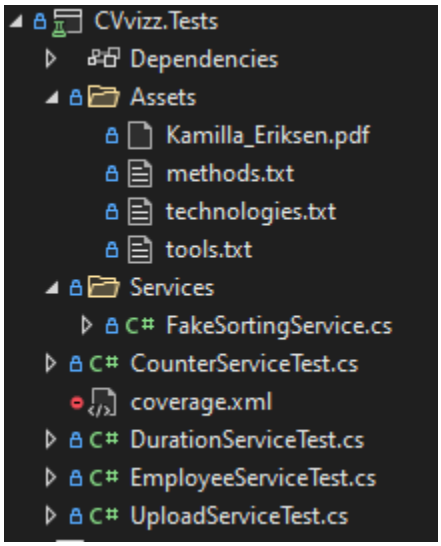




Figur nr. 9 og 10: *Serverens mer detaljerte prosjektstruktur.*

3.4 Prosjektstruktur til test prosjektet

Prosjektstrukturen til test prosjektet består av *Assets, Services* med en falsk service samt de ulike test klassene.

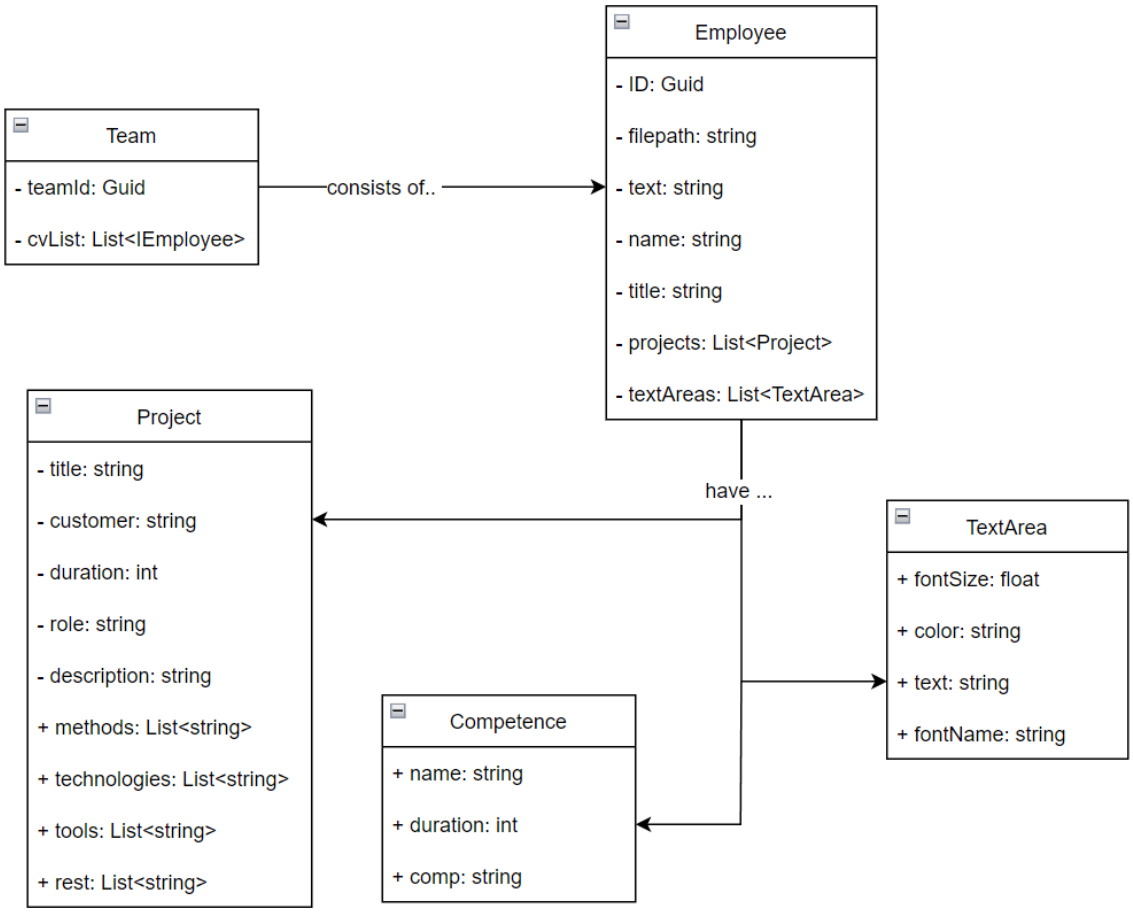


Figur nr. 11: *Prosjektstruktur til testprosjekt.*

4. Klassediagram

Vi så det som mest hensiktsmessig å bare utforme klassediagram for serveren på et overordnet nivå da klienten ikke håndterer objektene direkte.

Et *team* består av flere *Employees*. *Employee* har lister med *TextAreas* og *Projects* basert på CVen sin. *Project* inneholder prosjektets tittel, kunde, varighet, den ansattes rolle samt prosjektbeskrivelsen. Videre inneholder *project* lister med oversikt over metoder, teknologier og verktøy benyttet i dette prosjektet som matcher standardiserte lister for hver kategori. De kompetanseområder fra CVene som ikke faller innenfor kategoriene lagres i listen *rest*. *Competence* fungerer som et *data transfer object* (DTO) i koden og inneholder navn på ansatt, spesifikk kompetanse (enten rolle, teknologi, verktøy eller metoder) samt den totale varigheten med denne kompetansen.

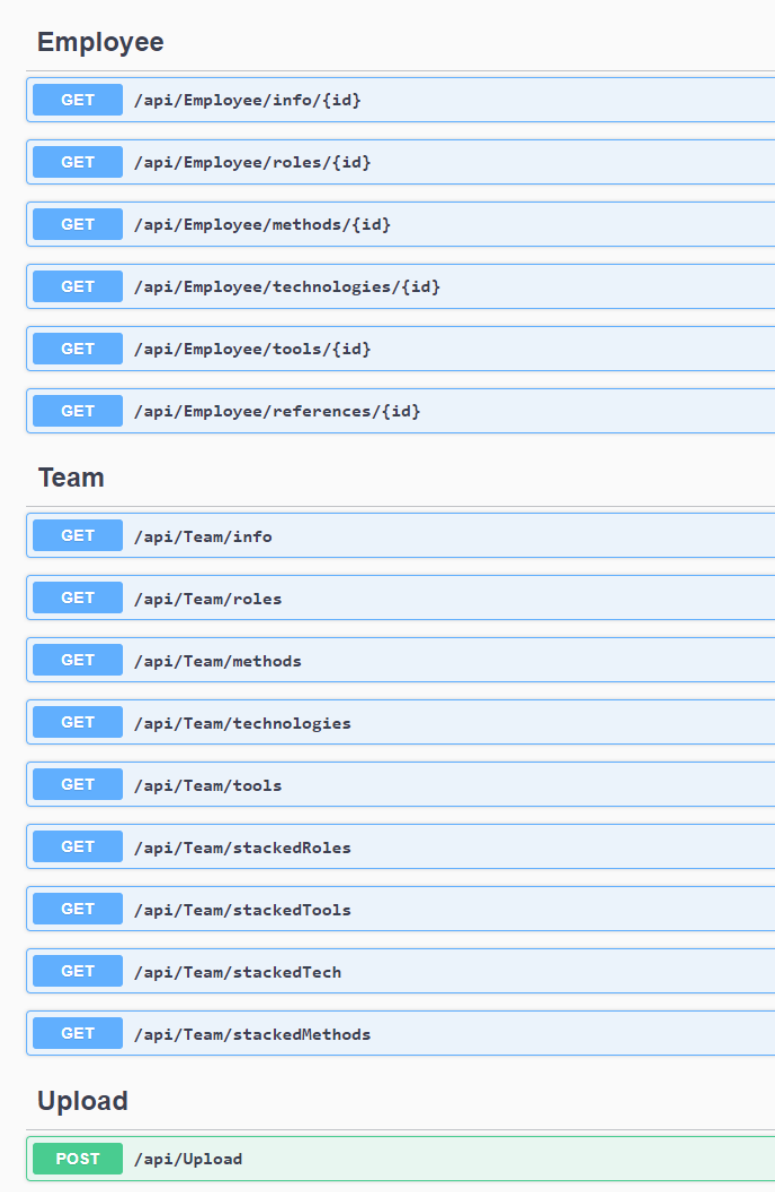


Figur nr. 12: *Datamodellene som benyttes.*

5. Server-tjenester

Når serveren til systemet kjører listes alle REST-ressursene på denne siden: https://localhost:5001/swagger/index.html

Employee har GET endepunkter for å hente ut navn, tittel, top fem referanser samt *dictionaries* med alle roller, alle metoder, alle teknologier og alle verktøy med deres tilhørende varighet. *Team* har de samme GET endepunkter bare for hele *teamet* samlet. Videre har *team* også fire GET endepunkter for de samme kategoriene bare på riktig format for stablet søylediagram. Formatet skal være kompetanseområdet først, deretter alle i *teamet* listet med deres respektive erfaring i mnd med den kompetanse. POST endepunktet /api/upload benyttes til opplastning av en til flere CV-er og returnerer de respektive ID-ene.



Figur nr. 13: *Endepunkter tilknyttet employee.*

6. Sikkerhet

Systemet benytter HTTPS-protokollen til kommunikasjon mellom klient og server. Det er tatt spesielle sikkerhetstiltak i forhold til opplastning av filer til server. For å sikre mot potensielt skadelige filer benytter systemet en whitelist, som utelukker alle andre filtyper enn PDF. Filen blir sjekket opp mot dette både i klient og i tjener. Systemet benytter ASP.NET sin egen øvre grense for filstørrelse (4 mb), og vil ikke godta filer større enn dette. Filens navn blir også endret i det den lagres i *temporary storage*, og dermed kan ikke en eventuell angriper benytte seg av filnavnet til skadelige handlinger. De viktigste sikkerhetsmekanismene er lagt på serversiden av systemet, slik at en angriper ikke kan foreta spørringer via andre kanaler enn klienten. Klienten er i tillegg avhengig av å ha et gyldig HTTPS-sertifikat utstedt av tjener for å kjøre spørringer mot endepunktene.

7. Installasjon og kjøring

Kildekoden ligger vedlagt til hovedrapporten under navnet CVvizz. Denne lastes ned lokalt og og kjøres via terminal. For å kjøre prosjektet må du ha installert:
Node.js - https://docs.npmjs.com/downloading-and-installing-node-js-and-npm.
.NET SDK x64 - https://dotnet.microsoft.com/en-us/download

Det inkluderer det som trengst for å bygge og kjøre systemet.

Kjøring av systemet skjer i to steg:

1. Først kjøres serveren opp ved å navigere til mappen *CVvizzbackend* i prosjektet og skriv kommandoen **'dotnet run'**.

2. Åpne deretter en ny terminal for å kjøre opp klienten. Navigere inn i mappen som heter *CVvizzfrontend* i prosjektet. Inne i mappen kjører du først **'npm install'** for å laste ned nødvendige moduler deretter **'npm run serve'** for å kjøre opp klienten.

Da er webapplikasjonen tilgjengelig på https://localhost:5002/ i ønsket nettleser.

For å teste applikasjonen er det vedlagt et utvalg test-CVer i mappen *Test-Data.* Disse er basert på eksisterende CV-er fra bedriften som er anonymisert grunnet personvern. Alle er blitt kvalitetssikret og godkjent av oppdragsgiver.

8. Dokumentasjon av kildekode

Kildekoden er dokumentert med XML dokumentasjon på serversiden og klientsiden med JSDoc. Fra prosjektstrukturen til serveren sees mappen *Documentation*, denne inneholder filen *CVvissbackend.xml* som inneholder en oppsummering av dokumentasjonen på serversiden.

9. Testing

Systemet har flere enhetstester og en integrasjonstest som tester opplastning av en CV. Til enhetstesting benyttes testverktøyet xUnit. Verktøyet kombineres videre med en lokalt lagret CV i pdf format for å gjennomføre en integrasjonstest. Til dokumentering av testdekningsgrad benyttes verktøyene AltCover og ReportGenerator på grunn av deres kompabilitet med systemet. Systemet har en testdekningsgrad på ca. 41.9% som kan sees fra rapporten i figur 14. Her sees at klassene ansvarlig for uthenting av informasjon fra CVene har høy dekningsgrad. Per nå har *controller* klassene, samt noen av de nyeste klassene ikke noe dekningsgrad som også kan sees i figur nr. 14.

For å kjøre testene navigerer man seg til mappet *CVvizz.Tests* og skriver kommandoen **'dotnet test'**.

– CVvizzbackend	453	628	1081	2193	41.9%	96	228	42.1%
<>f__AnonymousType0<T1, T2, T3, T4>	0	0	0	0		0	0	
CVvizzbackend.Assets.FontSizeSimpleTextExtractionStrategy	24	0	24	50	100%	2	4	50%
CVvizzbackend.Controllers.EmployeeController	0	62	62	138	0%	0	14	0%
CVvizzbackend.Controllers.TeamController	0	36	36	86	0%	0	0	
CVvizzbackend.Controllers.UploadController	0	30	30	70	0%	0	10	0%
CVvizzbackend.Models.Competence	0	20	20	49	0%	0	0	
CVvizzbackend.Models.Employee	41	6	47	104	87.2%	0	0	
CVvizzbackend.Models.Project	24	61	85	143	28.2%	0	8	0%
CVvizzbackend.Models.Team	0	24	24	54	0%	0	2	0%
CVvizzbackend.Models.TextArea	7	3	10	32	70%	0	0	
CVvizzbackend.Services.CounterService	37	190	227	403	16.2%	7	66	10.6%
CVvizzbackend.Services.DurationService	40	0	40	81	100%	4	4	100%
CVvizzbackend.Services.EmployeeService	7	48	55	121	12.7%	0	4	0%
CVvizzbackend.Services.ExtractionService	100	0	100	191	100%	36	38	94.7%
CVvizzbackend.Services.LocatorService	88	4	92	170	95.6%	30	34	88.2%
CVvizzbackend.Services.ParseService	37	0	37	80	100%	10	10	100%
CVvizzbackend.Services.ProjectService	30	0	30	75	100%	6	6	100%
CVvizzbackend.Services.SortingService	0	64	64	130	0%	0	22	0%
CVvizzbackend.Services.TeamService	0	41	41	100	0%	0	2	0%
CVvizzbackend.Services.UploadService	18	1	19	58	94.7%	1	2	50%
Microsoft.CodeAnalysis.EmbeddedAttribute	0	0	0	0		0	0	
Program	0	38	38	58	0%	0	2	0%

Figur nr. 14: *Rapport av testdekning generert ved hjelp av ReportGenerator og AltCover.*

Criteria	WCAG Version	WCAG Level	Recommendation	Conformance	Notes
1.1.1 Non-text Content	2	A	All img tags must have alt attributes.	N/A	
1.1.1 Non-text Content	2	A	If short alt text is sufficient to describe an image, provide the short text via the image's alt attribute.	N/A	
1.1.1 Non-text Content	2	A	If a short text alternative is not sufficient to describe an image (such as in a chart, graph, or diagram), provide short text via the image's alt attribute, and include a long description in nearby text.	N/A	
1.1.1 Non-text Content	2	A	If an image or icon is used as a button or link, the image has a text alternative sufficient to describe the purpose of the button or link.	N/A	
1.1.1 Non-text Content	2	A	Images that are decorative, used for formatting, or contain content already conveyed in text have a null alt attribute or are implemented as CSS background images.	Supports	
1.1.1 Non-text Content	2	A	Frames and iframes have descriptive titles.	N/A	

1.1.1 Non-text Content	2	A	Minimize the number of adjacent links to the same destination by combining adjacent images and text into a single link, rather than creating a separate link for each element.	Supports	
1.2.1 Audio-only and Video-only (Prerecorded)	2	A	For pre-recorded audio (without video), provide a descriptive transcript that includes dialogue and all other meaningful sound.	N/A	
1.2.1 Audio-only and Video-only (Prerecorded)	2	A	For pre-recorded video (without audio), provided a text alternative or audio descriptions that provide the same information presented	N/A	
1.2.2 Captions (Prerecorded)	2	A	Provide captions for prerecorded audio content in non-live synchronized media.	N/A	
1.2.3 Audio Description or Media Alternative (Prerecorded)	2	A	For non-live video, provide a descriptive transcript or an audio description.	N/A	
1.2.4 Captions (Live)	2	AA	Provide captions for live audio and video.	N/A	

1.2.5 Audio Description (P	2	AA	Videos should include "radio style" narration so that content makes sense if someone is consuming just the audio track. Include any text elements in the narration.	N/A	
1.3.1 Info and Relationships	2	A	Use semantic markup to designate headings, lists, figures, emphasized text, etc.	Supports	
1.3.1 Info and Relationships	2	A	Organize pages using properly nested HTML headings.	Supports	
1.3.1 Info and Relationships	2	A	Use ARIA landmarks and labels to identify regions of a page.	Does not support	
1.3.1 Info and Relationships	2	A	Reserve tables for tabular data, use table headers appropriately, and use table captions.	Supports	
1.3.1 Info and Relationships	2	A	When the appearance of text conveys meaning, also use appropriate semantic markup.	N/A	
1.3.1 Info and Relationships	2	A	Avoid emulating links and buttons. Use the a and button tags appropriately. Avoid using a tags for buttons. Avoid using div, span, etc. tags for links or buttons.	Supports with exceptions	Ikke mulig å unngå ved filopplastning

1.3.1 Info and Relationships	2	A	Avoid using whitespace characters for layout purposes.	Supports	
1.3.2 Meaningful Sequence	2	A	Ensure that the source order presents content meaningfully. When the page is viewed without styles, all content on the page should still appear in a meaningful and logical order.	Supports	
1.3.3 Sensory Characteristics	2,1	A	Do not identify content based on its color, size, shape, position, sound, or other sensory characteristics.	Supports	
1.3.3 Sensory Characteristics	2,1	A	Do not convey information solely through icons or symbols.	Supports with exceptions	
1.3.4 Orientation	2,1	AA	All content and functionality should be available regardless of whether a mobile device is oriented vertically or horizontally, unless the orientation of the device is absolutely essential.	Does not support	

1.3.5 Identify Input Purpose	2,1	AA	If a form field asks for information about the user and if there is an appropriate HTML autocomplete attribute, include that autocomplete attribute.	N/A	
1.4.1 Use of Color	2	A	Links should always be easily identifiable through non-color means, including both default and hover states. The easiest and most conventional way to signify links is underlining.	N/A	
1.4.1 Use of Color	2	A	Required fields and fields with errors must include some non-color way to identify them.	N/A	
1.4.1 Use of Color	2	A	When the color of words, backgrounds, or other content is used to convey information, also include the information in text.	N/A	
1.4.2 Audio Control	2	A	Do not have audio that plays automatically on the page. When providing audio, also provide an easy way to disable the audio and adjust the volume.	Supports	

1.4.3 Contrast (Minimum)	2	AA	Text (including images of text) have a contrast ratio of at least 4.5:1. For text and images of that is at least 24px and normal weight or 19px and bold, use a contrast ratio that is at least 3:1.	Supports	
1.4.4 Resize text	2	AA	Ensure that there is no loss of content or functionality when text resizes.	Supports	
1.4.4 Resize text	2	AA	Define texts and text containers in relative units (percents, ems, rems) rather than in pixels.	Does not support	
1.4.5 Images of Text	2	AA	Avoid images of text, except in cases such as logos.	Supports	
1.4.10 Reflow	2,1	AA	Provide responsive stylesheets such that content can be displayed at 320px wide without horizontal scrolling. (Content which must be displayed in two dimensions, such as maps and data tables, may have horizontal scrolling.)	Supports	
1.4.11 Non-text Contrast	2,1	AA	Color contrast for graphics and interactive UI components must be at least 3:1 so that different parts can be distinguished.	Supports	

1.4.11 Non-text Contrast	2,1	AA	When providing custom states for elements (e.g. hover, active, focus), color contrast for those states should be at least 3:1.	Supports	
1.4.12 Text Spacing	2,1	AA	Avoid using pixels for defining the height and spacing (e.g. height, line height, etc) of text boxes.	Supports with exceptions	
1.4.13 Content on Hover or Focus	2,1	AA	For tooltips, follow corresponding ARIA authoring practice.	Does not support	
1.4.13 Content on Hover or Focus	2,1	AA	For content that appears on hover and focus: the content should be dismissible with the escape key; the content itself can be hovered over; and the content remains available unless it is dismissed, it is no longer relevant, or the user removes hover and focus.	Does not support	
1.4.13 Content on Hover or Focus	2,1	AA	To the extent possible, content that appears on hover or focus should not obscure other content, unless it presents a form input error. or can be dismissed with the escape key.	Supports	

2.1.1 Keyboard	2	A	Avoid implementing access keys. When access keys and other keyboard shortcuts are implemented, they must not interfere with existing browser and screen reader provided shortcuts.	Supports	
2.1.1 Keyboard	2	A	All functionality should be available to a keyboard without requiring specific timing of keystrokes, unless the functionality cannot be provided by a keyboard alone.	N/A	
2.1.1 Keyboard	2	A	Avoid relying exclusively on pointer-driven events, such as onmouseover, to provide functionality when scripting. Generally, such functionality will also require scripting for keyboard operability.	Supports	
2.1.1 Keyboard	2	A	In general, avoid using scripts to remove focus from an element until the user moves focus manually.	Supports	

2.1.2 No Keyboard Trap	2	A	Ensure keyboard focus is never trapped on an element without an obvious way to move focus out of the element. Make sure the user can move focus to and from all focusable elements using a keyboard only.	Supports	
2.1.4 Character Key Shortcuts	2,1	A	If a keyboard shortcut uses only letter (including upper- and lower-case letters), punctuation, number, or symbol characters, then the user must be able to disable the shortcut, remap the shortcut, or limit the shortcut to only when a particular interactive element has focus.	N/A	
2.1.4 Character Key Shortcuts	2,1	A	If a keyboard shortcut uses only letter (including upper- and lower-case letters), punctuation, number, or symbol characters, then the user must be able to disable the shortcut, remap the shortcut, or limit the shortcut to only when a particular interactive element has focus.	N/A	

2.2.1 Timing Adjustable	2	A	Do not require time limits to complete tasks unless absolutely necessary. If a time limit is necessary, the time limit should be at least 20 hours, or it can be extended, adjusted, or disabled.	Supports	
2.2.2 Pause, Stop, Hide	2	A	Items on the page should not automatically move, blink, scroll, or update, including carousels. If content does automatically move, blink, scroll, or update, provide a way to pause, stop, or hide the moving, blinking, scrolling, or updating.	Supports	
2.3.1 Three Flashes or Below Threshold	2	A	Do not provide any content that flashes more than three times in any 1-second period.	Supports	
2.4.1 Bypass Blocks	2	A	Provide a link to skip to the main content as the first focusable link on the page.	N/A	
2.4.2 Page Titled	2	A	Make sure each web page has a title tag that is descriptive, informative, and unique.	Does not support	

2.4.3 Focus Order	2	A	Create a logical tab order through links, form controls, and interactive objects.	Supports with exceptions	
2.4.3 Focus Order	2	A	When inserting content into the DOM, insert the content immediately after the triggering element, or use scripting to manage focus in an intuitive way. When triggering dialogs and menus, make sure those elements follow their trigger in the focus order in an intuitive way. When content is dismissed or removed, place focus back on the trigger.	N/A	
2.4.3 Focus Order	2	A	Avoid using tab index values greater than 0.	N/A	
2.4.4 Link Purpose (In Context)	2	A	The purpose of each link can be determined from the link text alone, or from the link text and the containing paragraph, list item, or table cell, or the link text and the title attribute.	Supports	

2.4.4 Link Purpose (In Context)	2	A	If the visible text alone is not sufficient to convey meaning, use advanced techniques to provide additional meaning, such as ARIA attributes, screen reader only text, or the title attribute.	N/A	
2.4.5 Multiple Ways	2	AA	Each website should include at least two of the following: a list of related pages; table of contents; site map; search; or list of all pages.	Does not support	
2.4.6 Headings and Labels	2	AA	Ensure that on each page, headings, landmark labels, and form labels are unique unless the structure provides adequate differentiation between them.	Supports	
2.4.7 Focus Visible	2	AA	Provide keyboard focus styles that are highly visible, and make sure that a visible element has focus at all times when using a keyboard. Do not rely on browser default focus styles.	Supports with exceptions	

2.5.1 Pointer Gestures	2,1	A	Do not require multipoint or path-based gestures (e.g. pinching, swiping, dragging) for functionality unless the gesture is essential to the functionality.	Supports	
2.5.2 Pointer Cancellation	2,1	A	Avoid triggering functionality on down-events, such as onmousedown. Use events such as onclick instead.	Supports	
2.5.2 Pointer Cancellation	2,1	A	If a function is triggered on an up-event (e.g. onmouseup), provide a way to abort or undo the function.	N/A	
2.5.3 Label in Name	2,1	A	The accessible name for a UI element must contain any visual label for the element. Accessible names for UI elements should match visual labels as closely as possible.	N/A	
2.5.4 Motion Actuation	2,1	A	Avoid activating functionality through motion (e.g. shaking a phone). If motion triggers functionality, provide a way to disable the motion trigger, and provide an alternative way to activate the functionality.	Supports	

3.1.1 Language of Page	2	A	Provide a lang attribute on the page's html element.	Does not support	
3.1.1 Language of Page	2	A	When a visual label is present for an interactive element (e.g. link or form control), the accessible name of the element should contain the visual label.	N/A	
3.1.2 Language of Parts	2	AA	If a portion of the page is in a different language, use the lang attribute on that part.	N/A	
3.2.1 On Focus	2	A	When the focus change, the page should not cause a change in page content, spawn a new browser window, submit a form, case further change in focus, or cause any other change that disorients the user.	Supports	

3.2.2 On Input	2	A	When a user inputs information or interacts with a control, the page should not cause a change in page content, spawn a new browser window, submit a form, case further change in focus, or cause any other change that disorients the user. If an input causes such a change, the user must be informed ahead of time.	Supports	
3.2.3 Consistent Navigation	2	AA	When components are repeated across web page, they should appear in the same relative order with regard to other repeated components on each web page where they appear.	Supports	
3.2.3 Consistent Navigation	2	AA	When a navigation menu is presented on multiple pages, the links should appear in the same order on each page.	Supports	
3.2.4 Consistent Identification	2	AA	When components have the same functionality across several web pages, the components are labeled consistently on each page.	Supports	

3.3.1 Error Identification	2	A	Programmatically indicate required fields using the required or aria-required attributes. Also, visually indicate required fields in the form's instructions or form labels. Do not indicate required fields for CSS alone.	N/A	
3.3.1 Error Identification	2	A	Make errors easy to discover, identify, and correct.	N/A	
3.3.1 Error Identification	2	A	Identify errors using aria-invalid.	N/A	
3.3.2 Labels or Instructions	2	A	Use semantic, descriptive labels for inputs. Visually position labels in a consistent way that makes associating labels with form controls easy. Do not rely on placeholder text in lieu of an HTML label.	Supports	
3.3.2 Labels or Instructions	2	A	Provide text instructions at the beginning of a form or set of fields that describes the necessary input.	Supports	
3.3.2 Labels or Instructions	2	A	When providing inline help text, use aria-describedby to associate the help text with the input.	N/A	

3.3.3 Error Suggestion	2	AA	If an input error is detected and if suggestions for correction are known, provide suggestions for fixing the submission.	N/A	
3.3.4 Error Prevention (Legal, Financial, Data)	2	AA	Provide easy ways to confirm, correct, or reverse a user action where a mistake would cause a serious real-world consequence (e.g. submitting financial data, entering into a legal agreement, submitting test data, or making a transaction).	N/A	
4.1.1 Parsing	2	A	Validate all page HTML, and avoid significant validation / parsing errors.	N/A	
4.1.2 Name, Role, Value	2	A	Avoid creating custom widgets when HTML elements already exist. For example, use a and button tags appropriately.	Supports with exceptions	
4.1.2 Name, Role, Value	2	A	When creating a custom interactive widget, consult the ARIA Authoring Practices Document. Use ARIA labels, descriptions, roles, states, and properties to expose information about the component.	Supports	

4.1.2 Name, Role, Value	2	A	Use ARIA to enhance accessibility only when HTML is not sufficient. Use caution when providing ARIA roles, states, and properties.	N/A	
4.1.3 Status Messages	2,1	AA	When conveying a status message, use ARIA live regions or ARIA alerts to announce the message to screen reader users.	N/A	

Blindheimsvik, Larsen & Jansrud

Visualisering av kompetanse

NTNU
Kunnskap for en bedre verden