KnowONE

En markedsplass for karriereveiledning

Jakob Olsrud Johansen

Martin Johansen

Magnus Hjelmen Kristiansen

Gruppe 16

Sluttrapport – Bachelorprosjekt 2021

OsloMet – Storbyuniversitetet

Institutt for Informasjonsteknologi

Postadresse: Postboks 4 St. Olavs plass, 0130 Oslo

Besøksadresse: Holbergs plass, Oslo

Telefon: 22 45 32 00

PROSJEKT NR.
16

TILGJENGELIGHET
Åpen

BACHELORPROSJEKT

HOVEDPROSJEKTETS TITTEL	DATO
KnowONE – En markedsplass for karriereveiledning	25.05.2021
	ANTALL SIDER / BILAG 73
PROSJEKTDELTAKERE Jakob Olsrud Johansen, s333753 Martin Johansen, s333727 Magnus Hjelmen Kristiansen, s333740	INTERN VEILEDER Roza Abolghasemi

OPPDRAGSGIVER	KONTAKTPERSON
KnowONE AS	Mi Le Hagen

SAMMENDRAG

KnowONE – En markedsplass for karriereveiledning er et bachelorprosjekt ved OsloMet – Storbyuniversitetet, gjennomført av gruppe 16 i samarbeid med startup-bedriften KnowONE AS. Målet for prosjektet er å utvikle en ny markedsplass for karriereveiledning.

KnowONE er utviklet som en webapplikasjon, der det er mulig å interagere med andre brukere, for å tilegne seg den kunnskapen som trengs før man skal studere eller ut i arbeidslivet. Brukerne skal kunne være anonyme, for å unngå å føle seg ukomfortable i jakten på god karriereveiledning.

Webapplikasjonen er utviklet i .NET Core og C# på backend, og TypeScript og Angular på frontend.

3 STIKKORD
Webapplikasjon
.NET Core
Responsivt design

Forord

Dette dokumentet representerer sluttrapporten for vår bacheloroppgave i informasjonsteknologi ved OsloMet - Storbyuniversitet våren 2021. Dokumentet inneholder og beskriver planleggingen, fremdriften og utviklingen av en webapplikasjon for startup selskapet KnowONE.

Vi vil spesielt takke Mi Le Hagen (CEO) for veiledning gjennom hele prosjektet samt anskaffelse av flere hjelpsomme mentorer. Vi vil også takke Magnus Le (Mentor) og Roza Abolghasemi (intern veileder) for god oppfølging og veiledning rundt tekniske spørsmål og grafiske diskusjoner. Grunnet Covid-19 foregikk kommunikasjonen over nett der vi hovedsakelig brukte Slack og hyppige møter på Google Meet, selv med disse begrensningene fikk vi rikelig med oppfølging og generell hjelp.

Dokumentet er optimalisert for digital lesning grunnet det at vi bruker flere lenker for å henvise til informasjon. Foruten om lenker og referanser kan dokumentet fint leses på papirform.

Innhold

Kapittel 1:

Presentasjon

Innhold

1.1 Innledning

Dette kapittelet er en presentasjon til bachelorprosjektet vårt. Vi kommer til å gi en kort presentasjon av prosjektgruppen, oppdragsgiver og deretter presentere selve bachelorprosjektet. Presentasjonen kommer til å gi innblikk i hva prosjektet går ut på, hvorfor vi skal utvikle prosjektet og hvilke forutsetninger vi som gruppe har for utviklingen.

1.1.1 Presentasjon av gruppen

Vi er en gruppe på tre studenter som har kjent hverandre helt siden fadderuken i 2018 da vi startet på studiet. Helt siden det har vi jobbet på oppgaver sammen og dermed blitt godt kjent med våre sterke og svake sider samtidig som vi har blitt et godt team som har et sterkt samarbeid. Under arbeidet med bacheloroppgaven har det vist seg utrolig nyttig at gruppen kjenner hverandre godt noe som gjør det mye lettere å dele oppgaver seg imellom.

Jakob Olsrud Johansen, s333753

Jakob studerer informasjonsteknologi på OsloMet-Storbyuniversitet og er gruppens leder. Jakob har vært interessert i teknologi gjennom hele livet sitt og har de siste årene fått en lidenskap innen frontend-utvikling. Han interesserer seg for brukervennlig og responsivt design, og har gjennom bachelorprosjektet hatt hovedansvaret for dette. Han trives best når han får vist ferdigheter innen spesielt HTML og CSS, men også innen blant annet Typescript. Jakob har hatt mest fokus på utseende og følelsen av å bruke produktene.

FIGUR 1 - JAKOB OLSRUD JOHANSEN

Magnus Hjelmen Kristiansen, s333740

Magnus studerer informasjonsteknologi på OsloMet-Storbyuniversitet. Magnus foretrekker å jobbe på backend og elsker å ha mange store oppgaver ovenfor seg slik at han bare kan fokusere på koden og utviklingen. Under prosjektet jobbet han fullstack, med fokus på community delen av hovedprosjektet samt mye annen ekstra funksjonalitet. Det har også vært Magnus sin oppgave å sette opp og administrere Azure skytjenesten.

FIGUR 2 - MAGNUS HJELMEN KRISTIANSEN

Martin Johansen, s333727

Martin studerer informasjonsteknologi på OsloMet-Storbyuniversitet. Han syntes frontend utvikling er spennende og brenner for å finne nye og innovative løsninger. Under arbeidet med bacheloroppgave har han jobbet fullstack, men har mot slutten av prosjektet fokusert mer på frontend.

FIGUR 3 - MARTIN JOHANSEN

1.1.2 Presentasjon av oppdragsgiver og bedrift

KnowONE er en startup-bedrift, som ble stiftet i 2021 underveis i arbeidet vårt med bachelorprosjektet. KnowONE har som hovedinteresse å hjelpe alle som har spørsmål rundt yrker eller forskjellige karrierer. Bedriften ble stiftet for å kunne vise et mer reelt syn på flere yrker, samtidig som at det skal være mulig for brukere å knytte forbindelser.

Det er ofte et stort diskusjonstema om hva en skal studere, hvor en skal studere og eventuelt hva en kommer til å jobbe med etter endt studie. KnowONE ønsker å gjøre det lettere for alle som eventuelt skal begynne på nye studier med å finne ut hva en jobb faktisk

innebærer og hvordan forskjellige personer har kommet seg dit. Dette skal bli mulig ved at brukere kan stille direkte spørsmål innenfor en gruppe om et yrke eller en jobbtittel, som senere kan bli besvart av en innenfor det yrket eller med den tittelen.

1.2 Faglige forutsetninger

For å kunne gjennomføre dette prosjektet, har vi vært avhengige av å bruke kunnskapen og erfaring vi har tilegnet oss gjennom ulike fag i løpet av studiet. Ved hjelp av disse fagene har vi kunne både lagt til rette en gjennomførbar prosjektplan, samt hatt tilgang til mye nyttig fagstoff vi har fått bruk for underveis i prosjektet. Disse fagene har gitt oss de nødvendige forutsetningene for bachelorprosjektet vårt:

Webprosjekt, DAFE1200

Webprosjekt var et fag alle på gruppen hadde sitt første semester. Dette var et fag som gav oss innsikt i hvordan utviklingen av nettsider foregår. Faget lærte oss stort sett det grunnleggende og hadde generelt få restriksjoner som gjorde at vi kunne være kreative og lære mye selv. Dette faget ga oss et godt bilde på hvordan en nettside kunne utvikles funksjonelt og designes.

Programmering, DAPE 1400

Programmering var et fag som viste oss hvordan objektorientert programmering foregår. Samtidig viste det oss hvorfor det er viktig å ha en god struktur på prosjektet du jobber på. Under arbeidet med bachelor fikk vi god nytte av kunnskapen vi fikk fra faget. Vi forsto fort at dersom vi skal kunne holde kontroll på filene vi jobber med kreves det en god struktur som opprettholdes. Det viktigste vi tok med oss fra programmering kan være kunnskap om god mappestruktur.

Programutvikling, DATA 1600

Programutvikling var et fag vi hadde på vårt andre semester som var en videreføring av 'introduksjon til programmering (DAPE 1400). Dette faget ga oss erfaring på hvordan det er å jobbe med et stort prosjekt over en lengre periode. Her lærte vi hvor viktig samarbeid og kommunikasjon innenfor gruppen er, samtidig som vi forstod at en god prosjektstruktur er kritisk.

Databaser, DATA 1500

Faget databaser hadde vi også i vårt andre semester og hadde sitt hovedfokus på både generell databaseteori og praktisk arbeid mot en database. Faget lærte oss om litt forskjellige måter en kan jobbe med en database på, men kanskje viktigst lærte det oss om hvordan vi kan sette opp en database. Under arbeidet med bacheloroppgaven var databasekunnskapen svært nyttig slik at vi kunne ha god struktur.

Systemutvikling, DAFE 2200

Systemutvikling var et fag vi hadde i vårt tredje semester. Her lærte vi hvordan vi kunne får bruk for use cases, versjons kontroll og generell testing, av disse er versjons kontroll desidert det vi fikk mest ut av og har vært viktigst under vårt arbeid. Det er også utrolig viktig i arbeidsmarkedet så det å få praktisk erfaring med det var utrolig viktig.

Menneske maskin interaksjon, ADSE 2100

Menneske Maskin Interaksjon (MMI) hadde vi i vårt tredje semester og skulle hjelpe oss med å forstå teorien bak design. Vi lærte å se forskjellen på gode og dårlige design samtidig som vi fikk et innblikk på hvordan selv gode design kunne konkludere i dårlige løsninger. Faget hadde også vektlagt hvor viktig fargebruk er for å inkludere alle brukere. Faget var til stor hjelp under arbeidet med design for nettsidene våre.

Datanettverk og skytjenester, DATA 2410

Datanettverk og skytjenester var et av fagene vi hadde i vårt fjerde semester. Vi lærte hvordan vi kunne sette opp store nettverk samtidig som vi lærte hvordan vi skulle allokere IP-adresser. Vi lærte viktig teori om hvordan data overføres og viktigheten med å sikre informasjonsflyt. Under prosjektarbeidet fikk vi spesielt nytte av kunnskapen vi lærte rundt http-protokoller og deres eventuelle svakheter.

Webapplikasjoner, ITPE 3200

Webapplikasjoner hadde vi i vårt femte semester. Dette faget lærte oss hvordan vi kunne bruke utviklingsrammeverk til å enklere lage en webapplikasjon. Vi fikk praktisk erfaring med å jobbe fullstack samtidig som vi fikk lære viktige konsepter for webutvikling.

1.3 Bakgrunn for oppgaven

KnowONE ønsker å gjøre karriereveiledning lettere og mer direkte slik at flere kan få et reelt syn på hvordan det faktisk er å jobbe innenfor et yrke. Det finnes stor usikkerhet i det å søke på en utdanning eller jobb som en ikke har erfaring i fra før, og dette er noe KnowONE ønsker å gjøre noe med.

1.3.1 Problemstilling

Det finnes utallige spennende jobber og yrker å velge mellom. Noen mer kjent enn andre. Kjennskapen til forskjellige yrker og arbeidsplasser kan variere basert på miljø og geografisk lokasjon. En bivirkning av dette er at vi kun vurderer yrker vi har kjennskap til (Ramberg, 2006, s.19). Det er derimot noen yrker som er kjent de fleste steder, oftest prestisjefylte yrker med høyt press på gode prestasjoner, for eksempel ingeniør, advokat, tannlege osv. Det har i senere tid blitt utviklet muligheter for å kunne lese om forskjellige yrkesretninger. I Norge brukes det flere nettsider aktivt for akkurat dette. Ulempen med disse nettsidene er at informasjon er relativt spredt, og en er nødt til å vite hva en ser etter for å kunne dra nytte av informasjonen. Derfor ønsker KnowONE å lage en sosial plattform der brukere kan samhandle med hverandre og dra nytte av hverandres erfaringer fra tidligere yrker og

studier.

1.3.2 Konsept

KnowONE er et nettsamfunn der helt vanlige mennesker bidrar til karriereveiledning ved å dele sin personlige erfaring av sitt yrke, eller studieretning. Målet med løsningen er å forenkle prosessen for å finne ut informasjon om ulike studieretninger eller karriereretninger. Det skal bli utviklet en sosial plattform slik at brukere kan få en mer personlig oppfattelse og et mer detaljert inntrykk av forskjellige yrker. Løsningen har som formål å fjerne 'mellommannen' slik at alle brukere kan forstå en bransje eller stilling akkurat som den er.

1.4 Beskrivelse av løsningen

1.4.1 Beskrivelse av løsning for innsamling av mail

Denne løsningen var den første webapplikasjonen vi skulle utvikle i prosjektet. Formålet med denne nettsiden var å introdusere fremtidige brukere til den sosiale plattformen. Løsningen skulle være rimelig enkel, ved å bare inneholde statisk informasjon og et skjema som kunne fylles ut med noe personlig informasjon. Dersom en bruker valgte å fylle ut dette skjemaet, ville de motta en velkomstmail og meldes på et nyhetsbrev.

1.4.2 Beskrivelse av løsningen for en sosial plattform

Løsningen skal tilby måter for brukere å samhandle med hverandre og hjelpe dem med å samle inntrykk fra forskjellige bransjer og yrker. Vi ønsker å gjøre brukere oppmerksomme på usensurerte erfaringer fra andre.

Ved registrering blir brukerne automatisk allokert inn i relevante bransjegrupper, avhengig av hva de selv har valgt ved registrering. Eksempelvis vil en person som registrerer sin bransjetilhørighet innen IT automatisk bli en del av "IT-gruppen". Videre blir brukeren spurt om å svare på 4 spørsmål. Her er formålet å gi oss brukerens subjektive mening om en

bransje. Disse svarene kan brukes senere til statistikk eller leses av andre brukere. Spørsmålene er:

1. Hva er din tittel / rolle i nåværende stilling?
2. Hva er det beste med din nåværende jobb / studieretning?
3. Hva er det mest utfordrende med din nåværende jobb / studieretning?
4. Hva skulle du ønske noen fortalte deg før du begynte i nåværende jobb / studie?

Hensikten med disse spørsmålene er å samle inn informasjon fra brukerne om hva deres individuelle usensurerte mening er om en bransje. Ved å samle inn disse meningene ønsker vi å kunne gi andre brukere inntrykk av erfaringer slik at de kan få et mer realistisk forhold til bransjen de undersøker. Formålet med nettsiden er på ingen måte å henge ut eller kritisere forskjellige bransjer eller arbeidsplasser, men heller muligheten for brukere å diskutere deres personlige erfaringer.

Heretter står brukerne fritt til å starte diskusjoner, kommentere innlegg eller abonnere på grupper. Ved dato for innlevering foreligger det ikke en konkret metode for å motivere brukere til å samhandle med hverandre, men dette er noe som skal implementeres etter hvert.

1.5 Rammebetingelser og begrensninger

Grunnet det at bedriften var under utvikling, forelå det ikke store begrensninger som var nødvendig å forholde seg til. Oppdragsgiver ønsket derimot å kunne lage en MVP (Minimum Viable Product). En MVP er et produkt som inneholder minst mulig funksjonalitet for at produktet skal kunne brukes og forberede brukere til å eventuelt betale for produktet i fremtiden. Ved bruk av en MVP, bruker bedriften lite tid og ressurser på å lage en businessmodell, noe som viste seg å være gunstig for både bedriften og utviklingen av produktet, på den måten at vi raskt kom i arbeid med å utvikle webapplikasjonen.

Kapittel 2:

Prosessdokumentasjon

Innhold

2.1 Innledning

I dette kapittelet kommer vi til å forklare om prosessen med å utvikle bachelorprosjektet, fra start til slutt. Vi kommer til å beskrive hvordan vi har jobbet, hvilke hjelpemidler vi har brukt og hvorfor vi har brukt disse hjelpemidlene.

2.2 Forhold

På grunn av coronapandemien har hele prosjektperioden foregått digitalt. Dette har selvfølgelig ikke vært et ideelt arbeidsforhold, da et gruppeprosjekt ofte er avhengig av et godt sosialt miljø. Likevel føler vi at vi har klart å gjøre det beste ut av situasjonen, med ukentlige møter og generelt hatt god kontakt gjennom hele prosjektperioden. Vi har klart å tilpasse oss hverandre, og utviklet et seriøst, men samtidig humorpreget, forhold. Gruppemedlemmene har som beskrevet tidligere i prosjektet samarbeidet gjennom hele studieløpet og hadde fra før av et godt forhold, men vi har også utviklet et godt forhold til både oppdragsgiver og mentorer.

2.3 Kravspesifikasjon

Kravspesifikasjonen beskriver de krav oppdragsgiver har for produktet som skal utvikles. Når kravspesifikasjonen er spesifisert, fungerer den som et styringsdokument for studentene gjennom hele prosjektperioden. Selv om kravspesifikasjonen beskriver krav til produktet, kan likevel gruppen finne ut at ting kan løses på en bedre måte, senere i prosjektet, noe som vil føre til at kravspesifikasjonen endres underveis.

Før vi startet utviklingen av prosjektet, måtte vi gå gjennom krav og ønsker for tjenesten vi skulle utvikle, sammen med oppdragsgiver. Oppdragsgiver hadde til å begynne med en liste over ting som måtte være med på MVP'en.

- Mulighet for å sette opp ulike karriereprofiler
    - Utdannelse
    - Arbeidserfaring
    - Tilleggsspørsmål
        - «Beste/verste med stillingen?»

        - «Hva skulle du ønske noen fortalte deg før du begynte?»
- Brukere skal kunne være i kontakt med hverandre.

Etter en diskusjon om disse kravene, kom vi frem til at disse kravene skulle kunne utvikles godt innenfor tidsrammen, og vi sa oss villige til å fylle på kravspesifikasjonen. Når kravspesifikasjonen var fastsatt, og godkjent av både oppdragsgiver og studenter, kunne vi starte utviklingen. Før utviklingen av prosjektet så kravspesifikasjonen slik ut:

- Det skal utvikles et nettsamfunn for karriereveiledning
- Det skal være mulig å opprette en brukerprofil og legge til utdanning, erfaring og svare på noen enkle spørsmål
- Det skal være mulig å bruke tjenesten selv om man ikke er innlogget, men for å få muligheten til å legge ut innlegg, stemme på innlegg eller kommentere må man være innlogget
- Det skal være mulig å poste innlegg eller kommentere som anonym
- Man skal kunne utforske forskjellige samfunn, og få oversikt over hvilken erfaring de som har postet/kommentert har
- Tjenesten skal være brukervennlig og skal designes for bruk på alle enheter

2.4 Prosjektverktøy

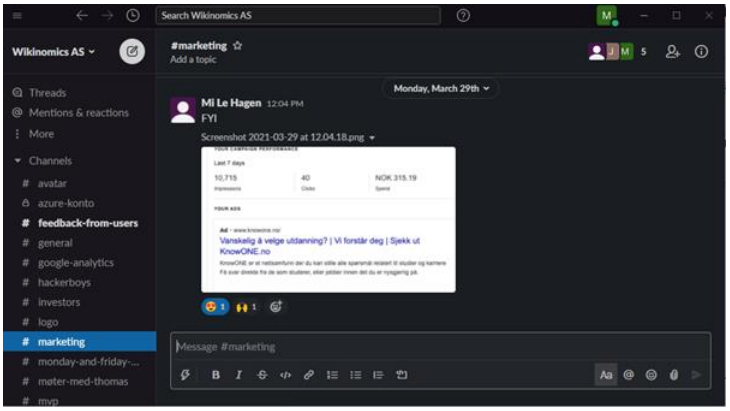
Gjennom utviklingen av prosjektet har det blitt tatt i bruk ulike verktøy, både på oppdragsgivers og gruppens initiativ. Disse ulike verktøyene har vært essensielle for utviklingen av prosjektet, da disse har gjort utviklingen og samarbeidet mellom gruppen og oppdragsgiver enklere. Disse verktøyene har vi brukt:

2.4.1 Samarbeidsverktøy

Utviklingen av prosjektet har hatt stort behov for samarbeidsvennlige verktøy. Med tanke på koronasituasjonen har vi ikke hatt noen form for fysisk oppmøte, så all informasjon og kommunikasjon har gått gjennom internett og diverse verktøy.

Slack

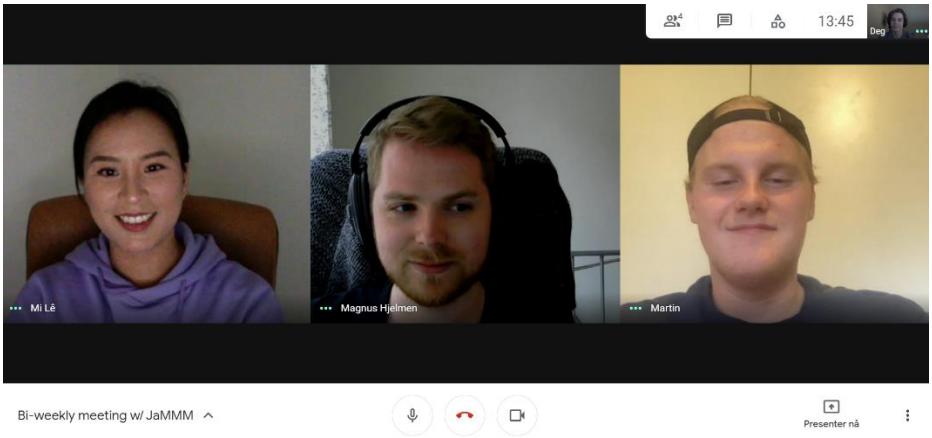
Slack er et av verktøyene som har blitt brukt mest under prosjektet. Slack er et program som gjør kommunikasjonen mellom brukere enkel. Muligheten til å opprette ulike kanaler gjør at diskusjoner og samtaler gjøres der det hører hjemme, istedenfor at alt er samlet på en plass. Stort sett all skriftlig kommunikasjon mellom oppdragsgiver og gruppen har foregått på Slack, og de ulike kanalene har gjort det enkelt å holde følge med hva som skjer.



FIGUR 4 - SKJERMBILDE FRA EN AV KANALENE PÅ SLACK

Google Meet

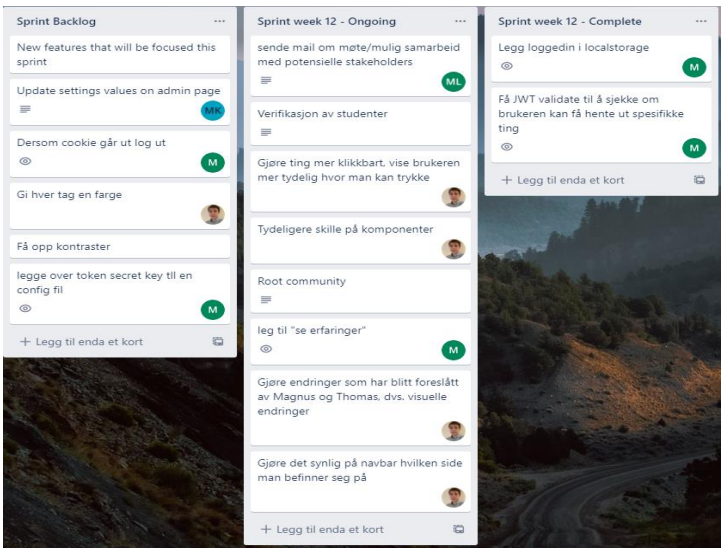
Gjennom utviklingsprosessen har vi hatt jevnlig møter med oppdragsgiver, i gjennomsnitt tre møter hver uke. Disse møtene har foregått gjennom Google Meet, da dette er den tjenesten vi hadde vår første samtale på og alle har følt seg komfortabel. Google Meet er en kommunikasjonstjeneste som gjør videosamtaler enkle å forholde seg til. Tjenesten åpner for enkel skjermdeling, noe som har gjort samarbeidet enklere, ved at endringer og nye ting som gjøres enkelt kan deles på skjermen for alle.



FIGUR 5 - MØTE MED OPPDRAGSGIVER

Trello

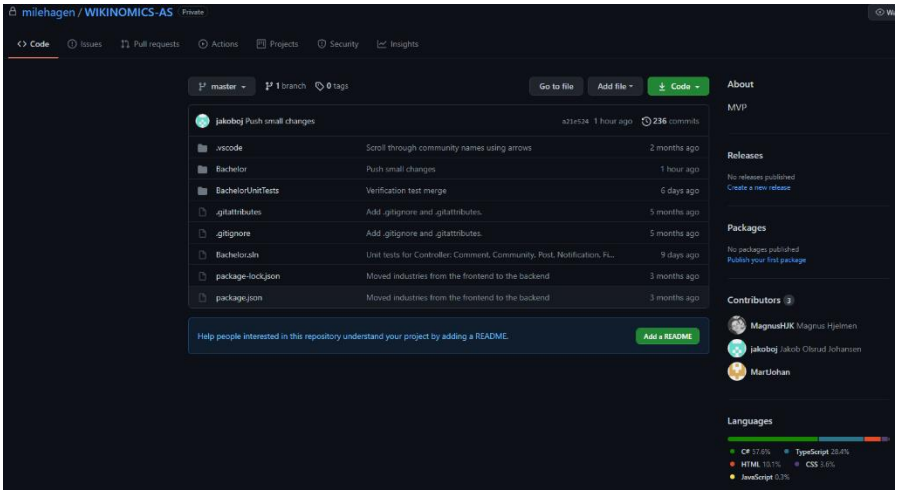
Det er viktig å holde orden på hva som har blitt gjort, hva som er under utvikling og hva som skal gjøres i et prosjekt. Trello er et samarbeidsverktøy som gjør dette enkelt å holde styr på. Ved å opprette såkalte «kort», kan man legge til hva som har blitt gjort, eller skal gjøres, og dette gjør det enklere for andre gruppemedlemmer å holde oversikt på hva status er for de andre.



FIGUR 6 - UTDRAG FRA VÅRT BRETT PÅ TRELLO

GitHub

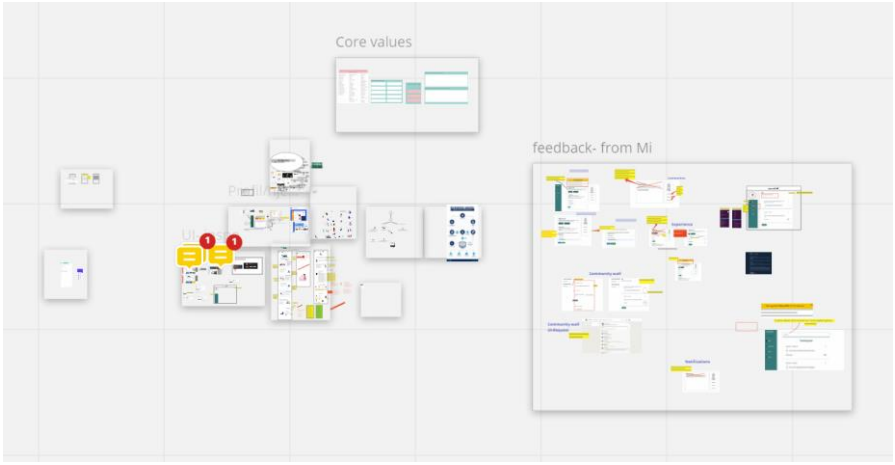
For å være oppdatert på hverandres arbeid har vi brukt GitHub. GitHub er en webapplikasjon som gjør det mulig å dele og hente ut andres kode ved hjelp av Git. Dette gjør det enkelt å passe på at man ikke overkjører andres arbeid eller jobber med noe som allerede er gjort.



FIGUR 7 - SKJERMBILDE FRA VÅRT GITHUB-REPOSITORY

Miro

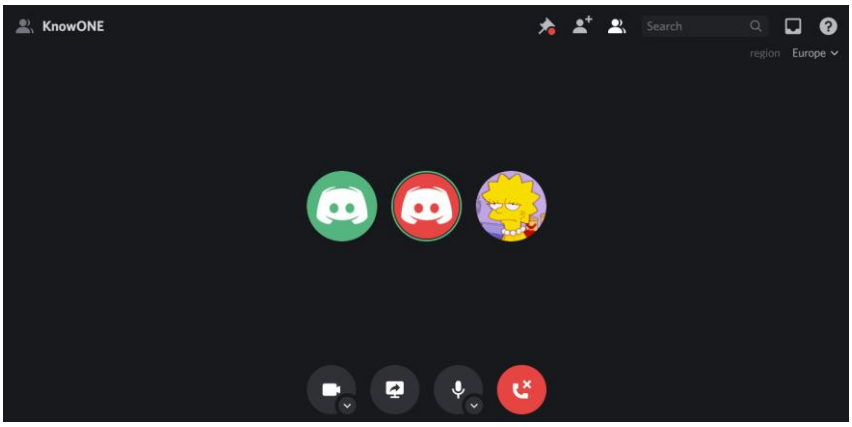
Miro er en samarbeidsplattform som vi ble introdusert for av oppdragsgiver. Dette er nesten som en lekeplass hvor man kan lage ulike skisser og leke seg med mulige design for prosjektet. Miro har særlig vært brukt når det kommer til utseende og innhold av både registeringssiden og selve tjenesten vi har utviklet. Tjenesten fungerer godt til samarbeid da man kan både se og redigere det andre medlemmer av rommet har gjort, samt legge til kommentarer.



FIGUR 8 - OVERBLIKK AV MIRO-BOARDET VÅRT

Discord

Discord har blitt brukt som en intern møteplass for gruppen. Discord er en plattform designet hovedsakelig for gaming, og fungerer som en blanding av Google Meet og Slack. Vi er alle godt kjent med Discord, og det har derfor vært naturlig å ha samtaler der når vi har hatt møter om ting som bare gjelder gruppen.



FIGUR 9 - GRUPPEMØTE PÅ DISCORD

Google Docs

For rapportskriving har vi samarbeidet i blant annet Google Docs. Grunnen til at vi har valgt nettopp Google Docs, er at dette er et tekstbehandlingssystem som er enkelt å samarbeide i. Vi har brukt Google

FIGUR 10 - GOOGLE DOCS

Docs under flere av rapportoppgavene våre, og dette er et system som alle i gruppen er kjent med.

FIGUR 11 - MICROSOFT WORD

Microsoft Word

Microsoft Word er tekstbehandlingssystemet vi har benyttet oss av for formatering av sluttrapporten. Her har vi blant annet designet prosjektforsiden og utformet rapporten slik vi ønsker at den skal se ut. Grunnen til at vi har valgt å formatere rapporten i Word fremfor Google Docs, er av erfaringsmessige årsaker. Vi har mer erfaring med oppsett i Word, og det er derfor naturlig at vi velger denne tjenesten for å utforme rapporten vår.

2.4.2 Utviklingsverktøy

Vi har under arbeidet med prosjektet også brukt ulike utviklingsverktøy. Her har det i stor grad vært mulig å velge verktøy ut ifra personlig preferanse, så lenge det ikke har ført til problemer i utviklingsprosessen.

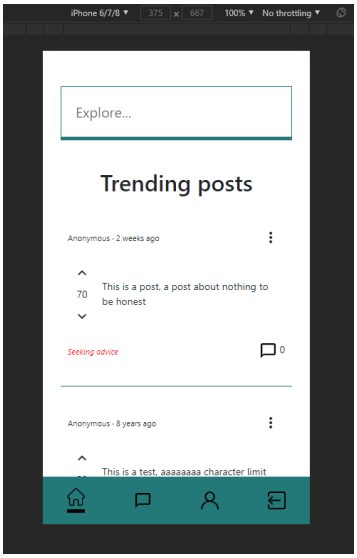
Visual Studio & Visual Studio Code

FIGUR 12 - VISUAL STUDIO CODE

Vi har brukt Visual Studio og Visual Studio Code som IDE gjennom prosjektet. Grunnen til at vi har brukt forskjellig IDE er på grunn av personlig preferanse, da Magnus ønsket en mer «komplett» IDE i Visual Studio, mens Martin og Jakob gikk etter brukeropplevelse i Visual Studio Code. Visual Studio er en IDE utviklet av Microsoft for utvikling i .NET, og kan kjøre .NET prosjekter uten nødvendige tilleggsinstallasjoner. Visual Studio Code, også utviklet av Microsoft, kan regnes mer som en Text Editor, men kan utføre den samme jobben som Visual Studio ved hjelp av extensions.

Google Device Toolbar

I en webapplikasjon er responsivt design viktig, som vil si at applikasjonen skal kunne brukes på alle enheter, uavhengig av størrelse. Google Device Toolbar er et innebygget verktøy i Google Chrome som befinner seg under «inspect», og gjør at man enkelt kan teste nettsiden hvor man befinner seg på forskjellige enheter. Verktøyet har vært essensielt for webapplikasjonens responsive design, da vi har kunnet teste tjenesten på for eksempel mobil gjennom nettleseren.



FIGUR 13 - GOOGLE DEVICE TOOLBAR

Microsoft Azure

I forbindelse med lansering av både landingssiden for registrering av mail og selve tjenesten vi har utviklet har vi brukt Microsoft Azure Web Apps. Med tanke på at vi har utviklet i .NET, har det vært enkelt å overføre løsningen vår til Azure skyløsning.

FIGUR 14 - MICROSOFT AZURE

2.5 Planlegging og metode

Scrum & smidig utvikling

Som vår arbeidsmetodikk valgte vi Scrum. Metodikken vektlegger fokus på flere iterasjoner, prioritere viktig funksjonalitet og levere god kvalitet på funksjonalitetene. Metodikken krever også at teamet er selvstyrte og at det ikke skal finnes en klar leder. En iterasjon i Scrum kalles for sprint og varer i cirka 1-4 uker. Vi valgte sprints med 1 uke, men dersom funksjonaliteten krevde mer tid utsatte vi den deretter. Scrum vektlegger også å få kjappe tilbakemeldinger slik at eventuell funksjonalitet kjapt kan endres før det sendes ut i produksjon. På slutten av sprint arrangeres det såkalte standups der hvor teamet viser frem det de har jobbet med i sprinten (Sommerville, 2015, s. 85).

Siden vi utviklet for en startup-bedrift, benyttet vi smidig utvikling slik at oppdragsgiver kunne se raske resultater (Sommerville, 2015, s. 76). Gjennom utviklingsprosessen endret kravspesifikasjonene seg og løsningen måtte tilpasses mer for brukerne.

2.5.1 Møter og endring i sprint-tidsramme

Allerede i vintermånedene 2020, før utviklingen av prosjektet startet, hadde vi videomøter med oppdragsgiver annenhver uke. I begynnelsen var disse møtene mest for å utforske ideer, bli kjent med hverandre, bygge tillit og gi et inntrykk av hva vi i gruppen kunne utarbeide over den gitte tidsrammen.

Når utviklingen av prosjektet begynte på nyåret, fortsatte vi med møter i samme format. Etter at vi ble ferdig med første del, introduksjonssiden til hovedprosjektet, bestemte vi oss for å kutte ned på tiden per sprint. Vi gikk fra to uker mellom hver sprint, til kun én, og valgte å ha et møte i starten av uken og et på slutten av uken.

Mandagsmøtet ble til et møte hvor vi kartla ukens sprint sammen med oppdragsgiver. På disse møtene fordelte vi arbeidsoppgaver til hvert enkelt gruppemedlem. Oppgavene ble utdelt ut ifra det vi i gruppa så på som viktigste prioritet, eller som oppdragsgiver ville ha

med. På disse møtene var det mye opp til hver enkelt å ta på seg oppgaver etter hva den enkelte følte den kunne få utført til fredag. Oppgavene ble da plassert i fellesskap på Trello.

Fredagsmøtene kalte vi for «demo day», da disse møtene gikk til å vise frem til oppdragsgiver hva vi gruppemedlemmer hadde jobbet med, og hva vi hadde fått til.

Hovedgrunner til nedkutting i sprint-tidsramme

1. Når alle gruppemedlemmene jobbet på hovedprosjektet samtidig, hadde vi nok ressurser til å øke hastigheten på utviklingsprosessen. Det ble derfor litt for lenge med to ukers sprints, og etter hvert som de store oppgavene som var nødvendige i begynnelsen av prosjektet ble ferdig og strukturen begynte å bli fastsatt, var det oppgaver av mindre omfang som stod fremfor oss.
2. Med to møter i uken var det lettere for oss å få et samlet perspektiv og se at konkrete oppgaver ble utført. Demodagen på fredager utgjorde også stor nytte for oppdragsgiver, som ikke har noen teknisk bakgrunn. Å kunne se en visuell og tydelig fremgang var behjelpelig for oppdragsgiver, ettersom ting da ble mer klart og vi kunne raskt få konkrete tilbakemeldinger
3. Tipset om to ukentlige møter og kortere sprints fikk vi fra vår mentor, Magnus, og Thomas, som var en av støttespillerne våre gjennom prosjektet. De sa dette kunne være lurt for å få enda bedre kommunikasjon, slik at alle var på samme side. Vår felles visjon for prosjektet skulle ikke være basert på antagelser fra hver enkelt part, men heller en felles forståelse av hva hver part ville oppnå.

Vi opplevde endringer på sprintvarighet og antall møter per uke som svært givende. Prosjektet fikk jevnere og tydeligere flyt, og dialogen med oppdragsgiver ble tydeligere. Samtidig følte vi at oppdragsgiver fikk en bedre forståelse for hva vi gjorde, og tiden det tok å utføre og implementere features.

2.6 Oppsummering etter uker

Uke 1 – Mailside-basisfunksjonalitet

Startet med å ha møter innad i gruppen for å snakke litt om løpet og hvilke tidlige tanker vi hadde. Senere hadde vi møte med oppdragsgiver og vi ble enige om at vi i første omgang skulle prioritere en landingsside.

Vi opprettet GitHub repositorier til både hovedprosjektet og til mailsiden. Arbeidet på mail siden startet kort tid etter. Funksjonalitet som innsamling av mail-adresser og admin-panel hvor utsending av mail til mailinglisten ble implementert, samt sikkerhet og inputvalidering som hører med slike funksjoner. Det ble laget et simpelt design for å vise frem siden, ettersom oppdragsgiver kommer til å komme med et komplett design ønske senere.

Uke 2 – Videre mailside, Azure og hovedprosjekt

Mye av funksjonaliteten for mailsiden var på plass og vi la det midlertidige prosjektet ut på Azure for å vise til oppdragsgiveren hvordan det ville se og føles ut. Dette gjorde det mulig for oppdragsgiveren å komme med tilbakemeldinger og gi uttrykk for hvordan de vil ha designet og utformingen.

Ekstra funksjonalitet som statistikk over mail-listen ble lagt til. Ettersom det hovedsakelig var design og innhold som stod igjen for mailsiden begynte to av gruppemedlemmene å sette opp mapper og annen struktur for hovedprosjektet. Dette slik at vi kunne kjøre et todelt løp, hvor to medlemmer kunne implementere enkle og tidlige funksjoner mens den tredje medlemmene kunne fokusere på design og utforming for mailsiden.

Uke 3 - Brukerregistrering og design

Designet på mailsiden blir stadig forbedret og utvidet ut ifra ønsker fra oppdragsgiver. Navbar og den generelle utformingen er kommet på plass. Mailsiden skal være relativt simpel i design og oppsett, hvor brukere blar seg nedover og prosjektet blir forklart ettersom man kommer seg nedover.

På hovedprosjektet kan en bruker nå registrere seg og bruker objekter blir da opprettet på backend.

Uke 4 – Communities og login

Side for communities blir opprettet. Dette skal være den delen av prosjektet hvor brukere skal kunne stille spørsmål og få svar innad i relevante grupper. Denne uken får vi på plass muligheten til å stille spørsmål til spesifikke communities, og muligheten til å så kommentere til et spørsmål.

RouterLink ble lagt opp slik at man skal kunne linke til en ønskelig post eller community.

En opprettet bruker skal nå også ha muligheten til å logge inn på siden, med nødvendig sikkerhet.

Uke 5 – JWT, oversatt mail side

På mailsiden er det nå mulig å velge mellom en norsk side eller en engelsk side, med innhold på begge disse språkene. Melder man seg opp til nyhetsbrev får man velkomstmail på språket man brukte når man skrev seg opp.

Arbeidet på å generere JSON Web Tokens blir startet, dette for å senere kunne verifisere brukere og hva de kan gjøre.

Det blir lagt til funksjonalitet til å stemme enten opp eller ned på en post/comment. Det er nå også mulig å «tagge» et innlegg med ferdigdefinerte tags som «Question», «Seeking advice» og lignende slik at folk kan få et innblikk i hva innlegg handler om.

Uke 6 – Utforming, cookies/JWT og stemmer

Mailsiden sitt design ble gjennomgått og arbeid ble gjort for å sørge for at utformingen ble mer universal. Det ble fokusert mye på at siden skulle se og oppføre seg responsivt på mobilplateformer.

Ved logg inn på hovedprosjektet blir det nå generert en cookie med JWT inni seg. Dette skal brukes for å holde styr på Tokens som blir laget for brukere og gi oss lett tilgang til dem senere. Fokuset ble mye på JWT denne uken for å forstå hvordan det kan brukes på en riktig og fornuftig måte.

Fikset i hvordan stemmer på innlegg og kommentarer fungerer. Nå blir alle stemmer og hvilken bruker som har stemt tatt vare på og lagret i databasen vår. På denne måten kan vi sørge for at brukere ikke stemmer dobbelt, og at stemmer i samme retning som før bare fører til at stemmen blir annullert.

Uke 7 – Sammenkoble brukere og communities

Brukere ble koblet sammen med communities, tidligere var det kun midlertidige og genererte brukere som kunne lage innlegg og kommentarer. Ettersom bruker objekter og communities har kommet til et punkt hvor de er mer komplette følte vi at det var på tide å knytte de sammen. Det ble også lagt til muligheten for å publisere innlegg og kommentarer anonymt. En bruker er fremdeles påkrevd og knyttet til det publiserte innholdet, men identiteten blir ikke fremvist.

Det er nå også mulig å rapportere et innlegg eller kommentar, håndtering for dette vil komme senere.

Bruk av en del API kall ble rensket opp i slik at API funksjoner ikke ble kalt unødvendig. Det ble også flyttet en del midlertidig sikkerhetsfunksjoner fra front-end til back-end der det egentlig hører hjemme.

Uke 8 – Domenesertifikasjon, utvidet sign up

Det blir gjort klart til å lansere mailsiden, i denne sammenheng blir det gjort mye arbeid på Azure oppsettet. Her brukte vi en del tid på å få et gratis sertifikat for domene og andre nødvendigheter.

Ved registrering av brukere på hovedprosjektet er det nå flere ting som en bruker må fylle ut, slik som hvilken bransje eller studieretning personen er innen. Dette er for å kunne plassere brukere i relevante communities.

Rapporterte innlegg og kommentarer kan nå bli håndtert i et midlertidig adminpanel. Her kan man bestemme om den rapporterte entiteten skal slettes eller bli stående.

Uke 9 – Brukere matchet med communities

En ny bruker med et valgt fagfelt eller studieretning blir nå automatisk plassert i matchende community. Disse industriene og fagfeltene ligger nå også korrekt på back-end og er knyttet til brukere.

JSON Web Tokens blir nå hentet ut fra cookies og kan bli dekodet på back-end for å finne hvilken bruker som er logget inn.

En basis profilside blir opprettet som senere skal innholde informasjon om brukere og hvilke communities de tilhører.

Uke 10 – Design og subscribe til communities

Designet på hovedprosjektet starter å bli oppdatert fra det helt grunnleggende det har vært frem til nå. Index siden og navbar blir lagt til. På index-siden blir det nå hentet ut «trending» innlegg. Dette er innlegg som skal være av særlig interesse for brukere som kommer inn på siden.

Brukere kan nå abonnere på forskjellige communities, slik at det blir lagt til i dine communities.

Deler av siden reflekterer nå også om du er logget inn eller ikke.

Uke 11

Designet på communities og innlegg begynte å bli definert og et felles oppsett og utforming kom på plass. Vi valgte å legge opp mer tomrom mellom elementer på siden slik at den skulle bli mer oversiktlig og lettere å manøvrere. Samtidig ble det lagt til en egen feed som skal være en samling av innlegg fra alle communities som en bruker har abonnert til, likt en personlig feed på Facebook eller YouTube. En del logikk for inn- og utlogging av brukere ble lagt til slik at disse to tilstandene og fasene overlappet hverandre mindre.

Uke 12

Vi la til funksjonalitet for å kunne svare til andre kommentarer i diskusjonstråder. En kommentar som svarer til en annen kommentar, vil ha en markert tekst øver som indikerer hvilken kommentar du refererer til. Trykker man på referansen, blir den refererte kommentaren markert. Dette for å skape en mer dynamisk og oversiktlig følelse i tråder, samt skape en mer interaktiv følelse mellom brukere. En navbar ble lagt til for å kunne forsikre enkel og oversiktlig manøvrering innad i prosjektet. Oppsettet av Azure for å legge ut prosjektet på skyene ble startet.

Uke 13 – Oppdatering av prosjekt og Azure

Prosjektet ble oppdatert til .NET 5 fra den tidligere .NET Core 3.1. Dette ble gjort ettersom en del av Nu-Get-pakkene våre foretrakk nyere .NET versjon. Oppdateringen skulle egentlig ha minimalt for oss som utviklere å si, dessverre skapte dette store problemer på Azure skytjenesten vår. Dette fordi i forrige uke ble en løsning eksportert på Azure som var av versjon Core 3.1. Når vi da hadde oppdatert prosjektet til versjon 5 og eksporterte det på nytt, ble det store feil på skytjenesten. Løsningen ble til slutt å sette opp en ny «App service» på Azure. Etter at problemet ble løst fullførte vi utrullingen av prosjektet på Azure. Dette for igjen å gi oppdragsgiveren vår et innsyn i hvordan løsningen ville se og føles ut.

Vi lagde også et design for profilsiden som gjenspeiler designet på resten av siden.

Uke 14

Brukere som registrer seg med erfaring eller studieretning kan nå få dette verifisert. Dette gjøres ved at man oppgir en e-post adresse, hvor man kan få tilsendt en mail og dermed bekrefte eierskap av mail kontoen. Mail kontoene må være på domener som vi gjenkjenner, slik at det ikke er mulig å bruke offentlige domener som gmail.com eller hotmail.com. Det skal være for domener som oslomet.no, som skal indikere at man har noen form for tilknytning til studiesteder eller bedrifter.

Et community kan nå også ha flere communities «under» seg, for å skape et hierarki av communities. Dette slik at det kan være øvre grupper som er brede og generelle, med mindre og mer spesialiserte grupper inn under den øvre gruppen.

Uke 15

Brukere kan nå velge hvilken identitet (navn eller anonym) og erfaring (arbeid eller studier) de vil vise frem når de legger ut innlegg eller kommenterer. Dette er for at brukere enkelt skal kunne vise frem informasjon de selv synes er relevante for en diskusjon eller spørsmål. Samtidig som andre brukere skal kunne se om innlegg og kommentarer kommer fra brukere med noe form for relevant bakgrunn.

Ved verifiseringprosessen av erfaring kan man nå be om å få et domene lagt til. Dette betyr at hvis du oppgir en e-mail konto du skal bruke til å verifisere erfaring, og denne kontoen har et domene vi ikke gjenkjenner skal brukere nå ha muligheten til å be om at domene blir lagt til. Disse domenene kommer nå opp i adminpanelet vårt og vi kan velge å godkjenne det eller ikke. Over tid vil tjenesten få inne mange domene forslag og dermed spare oss for tid i å manuelt oppsøke domener og legge til disse.

Uke 16

Denne uken ble brukt til å gjøre deler av siden mer dynamisk og responsive. Dette ble gjort ved å gjøre flere kall til backend-asynkrone. Vi kunne dermed endre verdier og innhold som ble vist så fort de ble endret. Et eksempel på dette er at man nå kan abnormere og av-abnormere fra communities og notifikasjoner og denne endringen blir visuelt synlig med engang. Det ble også utført smålig fiksing på design for å forbedre brukeropplevelsen.

Det er nå også mulig å abonnere til å få notifikasjoner for tråder. Da vil en bruker få beskjed om det har skjedd en utvikling i en tråd de har abonnert til. Dette er gjort i et forsøk på å få brukeren til å komme tilbake og interagere mer med systemet.

Uke 17 – Avsluttende arbeid

Designet for utfylling av erfaring under profilsiden ble endret og gjort enklere for brukeren å manøvrere og forstå.

Fra dette punktet og til slutten av prosjektperioden ble mesteparten av tiden brukt på å lage tester for kontrollerne, fikse bugs i koden, og skrive på sluttrapporten.

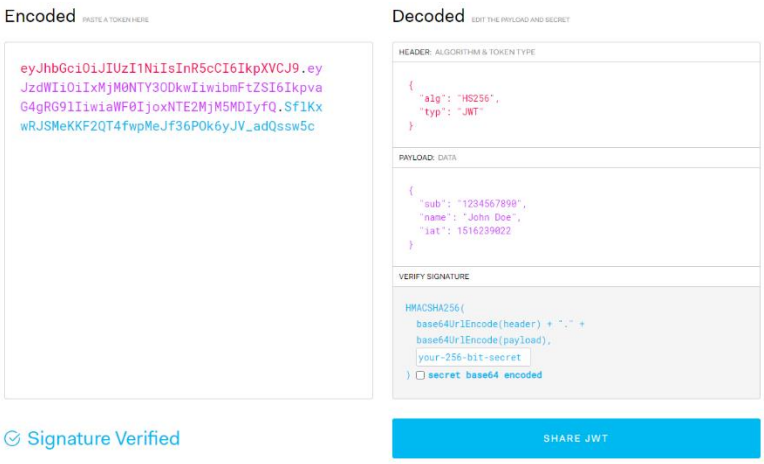
2.7 Faglige utfordringer

Med et prosjekt av så omfattende omfang som et bachelorprosjekt vil det alltid være utfordringer. For oss har det vært flere områder som har fremstått som utfordrende.

JWT

JSON Web Token var teknologien vi valgte å bruke for å autentisere brukere, verifisere hvilke API-kall brukerne kunne utføre og hvilken informasjon de skulle ha tilgang til. Genereringen av tokens, og dekodingen av tokens som brukeren sendte tilbake ved kall til backend var operasjoner og teknologier vi aldri hadde jobbet med og derfor brukte noe tid på å få implementert. Utfordringene kom hovedsakelig rundt den generelle forståelsen av hvordan

vi skulle gå frem for å lage tokens på en sikker måte og bruke dem effektivt. Vi brukte en online debugger vist i figur 15 for å teste om våre tokens var godkjente (JWT, u. å.).



FIGUR 15 - DEBUGGER

Asynkronekall og deling av data på front-end

I en webapplikasjon med mye data fra databasen er et sentralt spørsmål hvordan man skal håndtere den asynkrone dataen og fordeling av dataen til komponenter som trenger den. I denne sammenheng brukte vi en del tid på å sette oss inn i det som blir kalt for en service i Angular-rammeverket. En service skal utføre HTTP kall til back-end og holde på dataen som kommer tilbake. Når dataen er blitt hentet kan komponenter som trenger tilgang, abnormere og dermed bruke dataen internt slik den trenger det.

Azure

Azure skytjenestene fra Microsoft er store og komplekse økosystemer. Det tok tid å manøvrere seg rundt i Azure portal og bli kjent med oppsettet. Det tok spesielt tid å sette opp SSL sertifikater for tjenestene våre. Vi måtte gå gjennom mange ukjente steg og prosesser som var unikt for knyttingen mellom Let's Encrypt og Azure, og som ikke var offisielt støttet.

Å beholde database modellen oppdatert fra utvikling til produksjon var til tider vanskelig, men ble mye lettere når vi ble kjent med metoder for å migrere modellen på en standardisert måte og dermed oppdatere databasen etter disse endringene.

Design

Ved flere elementer i hovedprosjekt er det blitt brukt Angular Material-komponenter som gjør oppgaver som nedtrekkslister og ikoner mye lettere å implementere (https://material.angular.io/). Det var dessverre litt vanskelig å endre små detaljer på designet til disse elementene, og tok lang tid å få oversikt over hvordan slike endringer kunne overskrive standard utformingen.

Universell utforming er et svært viktig prinsipp når man designer og konstruerer web applikasjoner, vi har derfor fra et tidlig punkt hatt stort fokus på det. Både mail-siden og hovedprosjekt har gått gjennom mange design endringer for å gjøre det lettest mulig å manøvrere og bruke sidene. Det tok lang tid å få dette til å fungere på best mulig måte, og vi er veldig fornøyde med at begge sidene fungerer bra på desktop samt mobilenheter.

2.8 Konklusjon til prosessdokumentasjon

Utviklingsprosessen har vært mye lenger enn det vi er vant til fra prosjekter fra studiet. Dette har vært utfordrende, men også veldig givende. Det har vært spennende å kontinuerlig ha sprinter med tydelige mål. Gjennom sprintene og det veldig nære samarbeidet med oppdragsgiveren har vi fått en god struktur og kommunikasjonen har bare blitt bedre og bedre innad i gruppen og med oppdragsgiver.

Det er fremdeles mange funksjoner som kan bli lagt til som ville fått plattformen til å være mer sammenlignbar med sosiale plattformer av høyere kompleksitet, som for eksempel Facebook. Samtidig føler vi selv at vi har klart å utvikle noe som har et godt grunnlag og er en solid MVP for oppdragiveren som de kan bruke videre. Vi er fornøyde med design og de funksjonene som er implementert og føler at vi gir ifra oss et produkt som har stor mulighet for videreutvikling.

Målet med produktet var å vise frem ideene og ønskene som oppdragsgiver hadde for en fremtidig karriereveiledningsplattform.

Utviklingen av og prosessen rundt produktet har vært av stor verdi for alle involvert. Vi har lært veldig mye om utvikling av Angular-webapplikasjoner med .NET backend og REST API'er, noe som er ekstremt relevante teknologier for tiden. Det som fremdeles fremstår for alle i gruppen er hvor mye vi har lært av den formelle prosessen rundt utviklingen. Her har vi lært mye om det å kommunisere med kunder og personer med tilnærmet null teknologisk forståelse. Det har gitt oss et realistisk innblikk i hvordan en eventuell fremtid som utviklere kan være når man jobber oppimot andre personer enn bare andre utviklere. Vi har ikke bare fått en dypere forståelse, men respekt for prosesser designet for å gjøre utviklingsprosessen lettere og mer oversiktlig, som sprinter med sprint reviews. Disse prosessene sammen med andre hjelpemidler som illustrasjoner har gjort det langt lettere å jevne ut den store forskjellen i teknisk kompetanse mellom gruppen og oppdragsgiver.

Kapittel 3:

Produktdokumentasjon

Innhold

3.1 Innledning

I dette kapittelet skal tjenesten vi har utviklet presenteres. Vi skal forklare hvorvidt tjenesten oppfyller kravspesifikasjonen og beskrive de ulike komponentene av tjenesten, og hvordan disse komponentene fungerer. Siden vi i prosjektet har utviklet både en informasjonsside for tjenesten og selve tjenesten, kommer produktdokumentasjonen til å omhandle begge deler. Det kommer til å legges mest vekt på selve tjenesten, da dette har vært hovedprioriteten under prosjektperioden.

3.2 Introduksjon til tjenesten

Siden oppdragsgiver er en startup-bedrift, uten noe tidligere arbeid/prosjekt å vise til, er våre tjenester det første som er utviklet i regi av bedriften. Oppdragsgiver ønsket, som forklart tidligere, en samfunnsplattform for karriereveiledning. Vi har hatt stort fokus på å oppfylle oppdragsgiver forhåndsdefinerte ønsker for tjenesten, men også at tjenesten skal være enkel i bruk og fungere like godt på alle enheter.

Produktet er designet med et mål om responsivt design. Det er særlig lagt fokus på desktop-versjonen og mobil-versjonen, da vi ser for oss at dette er en tjeneste som kommer til å bli mye brukt på mobil. Det har derfor vært viktig for oss å legge fokus på at mobil-versjonen fungerer like godt som desktop-versjonen, og funksjonaliteten skal være den samme, selv om utseende er litt annerledes.

Målet med webapplikasjonen vi har utviklet er å gjøre det enklere for unge å få den informasjonen de både trenger og ønsker før de skal velge karriereretning. Det skal være mulig å kommentere både som offentlig og som anonym, men man skal uansett være trygg på at man får svar fra personer med relevant erfaring innen temaet. Man skal ikke føle seg redd for å bli dømt når man er på jakt etter svar, og dette er det KnowONE har blitt skapt for å forhindre.

3.3 Samsvar mellom kravspesifikasjon og produkt

Gjennom hele prosjektperioden har kravspesifikasjonen vært det viktigste styringsdokumentet vi studenter har hatt. Her har målene for prosjektet vært definert, og det er dette vi har strukket oss etter. I henhold til kravspesifikasjonen oppfyller webapplikasjonen vi har utviklet alle punktene. Det er mulig å opprette bruker, poste innlegg, kommentere og redigere egen profil, som oppdragsgiver og vi studenter så på som de viktigste punktene. Selv om vi har oppfylt alle kravene, er det fortsatt stort rom for videreutvikling etter vi er ferdige med bachelorprosjektet.

Vi har også underveis i prosjektet både lagt til ting på kravspesifikasjonen og fjernet ting. Grunnen til dette har vært at ting kan ha blitt mer utfordrende enn forventet, eller at enkelte deler har gått mer problemfritt enn vi først så for oss.

Oppdragsgiver ønsket i utgangspunktet at brukere skulle kunne ta kontakt med hverandre èn-til-èn, gjennom en chatfunksjon. Dette ble deprioritert, da oppdragsgivers hypotese er at terskelen for å starte en èn-til-èn-samtale er høyere enn terskelen for å anonymt delta i en samtale som foregår på et offentlig forum/nettsamfunn. Siden hensikten med MVP-en er å finne ut om folk er villige til å bruke tid på å dele erfaringer med fremmede, ønsket oppdragsgiver å senke enhver barriere som kunne oppstå. Derfor ble chatfunksjonen deprioritert, og fokuset på en åpen/felles vegg der brukere kan poste og kommentere ble prioritert.

3.4 Programmets oppbygning og virkemåte

3.4.1 Frontend

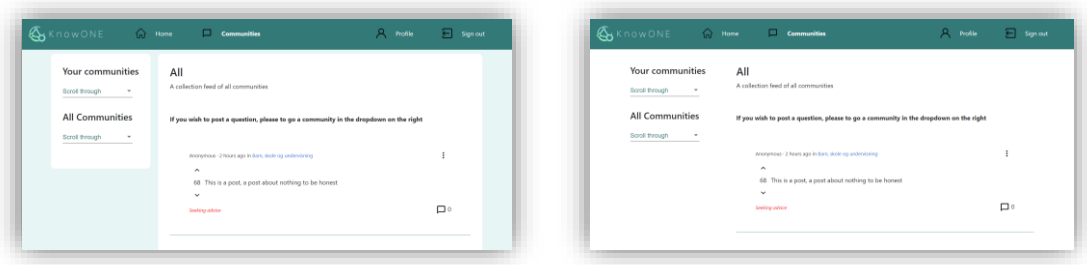
Gjennom prosjektet har det vært stort fokus på frontend-utvikling. Vår viktigste oppgave har vært å hjelpe oppdragsgiver med å nå frem til folk med sin idé. Derfor har det vært viktig å passe på at programmene våre både er brukervennlige, og enkle i bruk. Vi har fått god veiledning av vår frivillige mentor, som har delt sin erfaring og kunnskap med oss.

Designvalg

I denne delen kommer vi til å beskrive de ulike designvalgene vi har tatt og hvorfor vi har gjort disse valgene. Med tanke på at vi utvikler en MVP, er ikke designet nødt til å være perfekt, men vår tjeneste blir brukernes første møte med KnowONE, så design blir uansett en viktig del. Vi har fulgt diverse designprinsipper, og samtidig forsøkt å holde designet enkelt. Det viktigste med tjenestens design er at brukeren forstår hvordan tjenesten skal se ut og fungere.

Gestaltlovene hjelper oss å forstå hvilken effekt det visuelle har å si for brukerne, og dersom man utnytter disse lovene vil sjansene for at den visuelle utformingen fungerer som forventet (Sandnes, 2018, s. 64). På bakgrunn av denne beskrivelsen har vi valgt å utnytte gestaltlovene i utformingen av produktene vi har utviklet.

En av gestaltlovene vi har utnyttet er loven om forgrunn og bakgrunn. Forvirring om plassering av deler i et brukergrensesnitt er uønsket, og vi har derfor lagt vekt på dette på tjenesten vår. Ved å gjøre bakgrunnen på siden en annen farge enn det som vi ønsker å fremheve, vil det bli enklere for brukeren å skille mellom bakgrunn og forgrunn. I figur 16 har vi endret på bakgrunnsfargen fra figur 17, slik at det skal være enklere for brukeren å skille mellom bakgrunn og forgrunn.

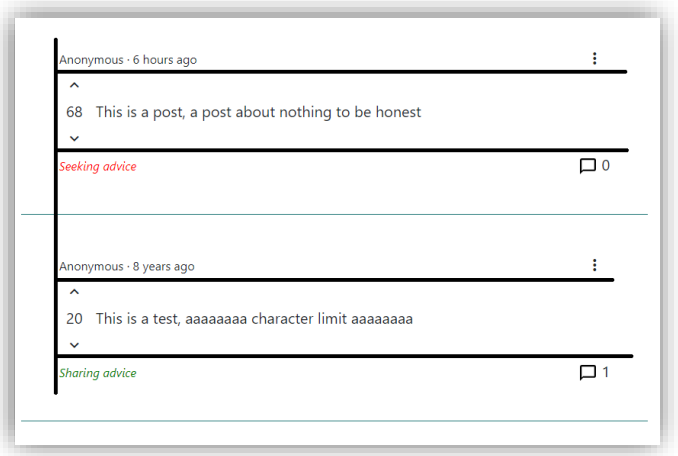


FIGUR 16 OG 17 – DEN LYSEBLÅ BAKGRUNNSFARGEN VIL FREMHEVE KOMPONENTENE MED HVIT BAKGRUNNSFARGE

Gruppering er også viktig i et brukergrensesnitt, for å skille mellom komponenter som ikke er i sammenheng med hverandre. Uten et skille mellom innlegg i feeden, kunne det vært enkelt å misforstå hva som burde sees i sammenheng. Derfor har vi valgt å skille alle innlegg med en grense. Et annet valg vi har tatt for å skille mellom ulike innlegg, er å endre

bakgrunnsfarge på innlegget som musepekeren svever over. Dette gir ikke bare brukeren et tydelig skille mellom innlegg, men gir også en følelse av klikkbarhet. Det er ikke nødvendigvis åpenbart for en bruker at det er mulig å klikke på et innlegg, og det er derfor vesentlig at brukeren gjøres oppmerksom på dette.

For å holde en form for struktur på tjenesten er kontinuitetsloven utnyttet. Kontinuitetsloven kan bidra til å fremheve struktur i en visuell fremstilling, ved å for eksempel følge usynlige linjer. Dette gjør at elementer som ikke er koblet sammen, kan føles tilhørende en gruppe, ved at de følger usynlige linjer (Sandnes, 2018, s. 73). Designet til innlegg og kommentarer følger kontinuitetsloven, ved å følge usynlige linjer. Selv om det ikke er noe fysisk skille, vil brukeren kunne oppfatte at vært innlegg er delt inn i tre rader, vist i figur 18.



FIGUR 18 - INNLEGG SOM FØLGER KONTINUITETSLOVEN

Illustrasjonene har vi blitt gitt fra oppdragsgiver, som er hentet fra Blush (https://blush.design/). Dette er en nettside hvor brukere får tilgang til illustrasjoner designet av utvalgte profesjonelle illustratører. I vårt tilfelle har vi hentet illustrasjoner fra kolleksjonen «Stuck at home», laget av Mariana Gonzalez Vega.

Oppdragsgiver har valgt illustrasjoner med sterke og 'lekne' farger som rød, turkis, gul og oransje. Ved bruk av disse fargerike illustrasjonene ønsket oppdragsgiver å skape et uformelt

miljø. Oppdragsgiver har gjennom premium-medlemskap kunne skreddersy utvalgte elementer etter ønske, og laste ned elementene i høy kvalitet.

Ikoner i produktene våre er laget av Freepik (https://www.freepik.com), og hentet fra Flaticon (https://flaticon.com).

Farger er viktige i et brukergrensesnitt. I våre produkter har vi brukt generelt få farger, men de fargene vi har brukt har ikke vært tilfeldig. Oppdragsgiver ønsket å bruke utvalgte farger fra illustrasjonene som utgangspunkt i KnowONE sin fargepalett. Utover i prosjektet har en mørk grønnfarge blitt symbolet på KnowONE's tjenester. For å holde designet enkelt og behagelig, har vi valgt å følge en monokromatisk fargeharmoni (Sandnes, 2018, s. 110).

Brukergrensesnittet følger også regelen om å kun benytte seg av tre hovedfarger (Slava Vaniukov, u. å.). Mørkegrønn, lyseblå og hvit er de tre fargene som går igjen gjennom hele tjenesten. Mørkegrønn i navbaren, lyseblå bakgrunn og hvit fremheving av interaksjonselementer.

For bachelorprosjektet vårt har en bekjent av oss, Sebastian Aaron Schwartz Johnsen, designet en logo, som vi har valgt å bruke i utviklingen vår og i rapporten. Om dette blir logoen KnowONE kommer til å ta i bruk etter at prosjektet er avsluttet er uvisst, men dette er altså logoen vi har valgt å bruke i vår rapport og tjenesten vi har utviklet.

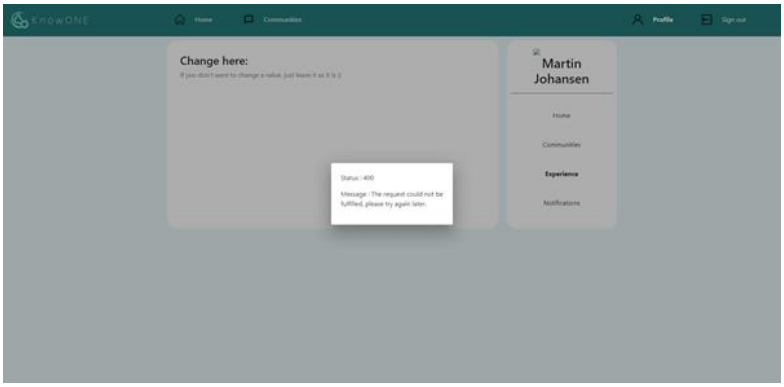
FIGUR 19 - LOGO

Brukerveiledning

I webutvikling er det viktig at brukergrensesnittet er forståelig. Det er ikke ønsket at en bruker skal være forvirret eller ikke forstå hva som kan gjøres i et brukergrensesnitt. For å unngå slike situasjoner, kan det være greit å hjelpe brukeren så godt som mulig, uten å spesifikt forklare hva som kan trykkes på, eller hva en knapp gjør. Vi har derfor gjort designvalg som skal gjøre brukergrensesnittet vårt enda enklere å forstå.

Dialogbokser

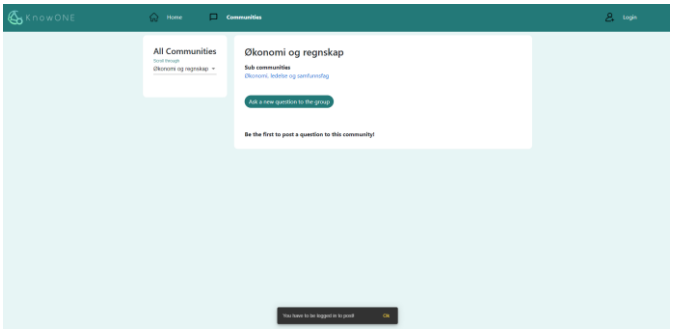
Dersom det forekommer et spesifikt http-error, skal brukeren bli varslet om dette. Vi har valgt å varsle brukeren for fire av http-kodene. Disse er 400, 401, 404 og 500. Http-kode 400 oppstår dersom det har forekommet en feil på klientsiden. Dette kan være at data som sendes til serveren er dårlig formatert. Http-kode 401 oppstår dersom en bruker prøver å aksessere informasjon som de ikke har tilgang til. Koden 404 oppstår dersom brukeren prøver å hente infromasjon som ikke finnes. Koden 500 oppstår dersom det har skjedd noe på serversiden. 500 er den meldingen som blir sendt dersom det ikke finnes en bedre kode for serveren å rapportere, så derfor valgte vi denne. Ved figur 20 vises det hvordan en slik dialogboks ser ut.



FIGUR 20 - DIALOGBOKS MED KODE 400

Snackbar

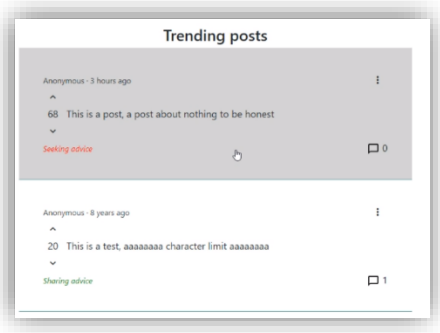
Som med mange komponenter i prosjektet vårt, brukte vi en Angular Materials-komponent kalt snackbar, for å gi brukeren tilbakemeldinger på handlinger de utfører på siden. Vi valgte snackbar fordi den integreres lett med Angular-prosjekter, og utformingen passer godt med andre Materials-komponenter. Samtidig kan vi fylle den inne med hvilken som helst tekst vi trenger, og den har muligheten til å bli lukket av brukeren, hvis de allerede har forstått beskjeden.



FIGUR 21 - SNACKBAR NEDERST PÅ SIDEN

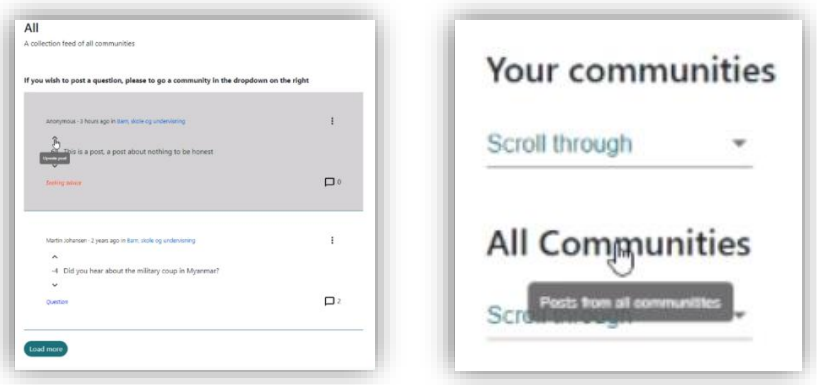
Klikkbarhet

For å gjøre det tydeligere for en bruker hva som er klikkbart har vi valgt å legge til små, men tydelige endringer i designet. Dette er endringer av for eksempel farge eller at musepekeren peker på elementer som det er mulig å klikke på, vist i figur 22.



FIGUR 22 - KLIKKBARHET VED ENDRET BAKGRUNN OG MUSEPEKER

Det er heller ikke alltid åpenbart hva en spesifikk knapp gjør, og det er derfor viktig å gjøre brukeren oppmerksom på nettopp dette. For en bruker som er helt ny til tjenesten og sosiale medier generelt, er det ikke åpenbart at en pil opp betyr at man kan avgi en stemme. Vi har derfor lagt til hjelpekommentarer, slik at brukeren skal få veiledningen som er nødvendig for at brukergrensesnittet skal gi mening. Eksempler på slike hjelpekommentarer vises i figur 23 og figur 24.



FIGUR 23 OG 24 - HJELPEKOMMENTAR FOR UPVOTE (TIL VENSTRE) OG SAMFUNN

Teknologier

Produktene vi har utviklet er en blanding av egenskrevet frontend-kode, og bruk av rammeverk, slik som Bootstrap. Bootstrap er et CSS-rammeverk, som gjør responsivt design enkelt (https://getbootstrap.com/). Bruk av Bootstrap gir ofte like design, som fjerner en del av den personligheten man kanskje ønsker på en nettside. Dette er en av grunnene til at vi har valgt å variere mellom bruk av Bootstrap og egenskrevet CSS- og TypeScript-kode.

Angular er et design-rammeverk og utviklingsplattform, egnet for 'Single Page Applications' (Angular, u.å.). Vi ble først introdusert til Angular høsten 2020, gjennom faget Webapplikasjoner, så med tanke på at vi allerede hadde faglige forutsetninger til å kunne utvikle en webapplikajson ved bruk av blant annet Angular, var valget av frontend-rammeverk enkelt.

3.4.2 Backend

Backend er på serversiden i klient-server-forholdet i en webapplikasjon. Her blir forespørsler tatt imot fra en klient og informasjon blir fremhentet fra databasen gjennom metoder som kontroller og et Data Access Layer.

Deler på backend

Kontroller

Kontrolleren tar imot forespørsel sendt fra klienten til backend. Disse forespørslene blir oversatt og sendt til DAL gjennom et interface, hvis de er av riktig format og autentisert ut ifra oppgaven som skal utføres. Når DAL har utført oppgaven sin, blir relevant data sendt tilbake til kontrolleren. Kontrolleren pakker da eventuelt dataen inn i et Object Result, og sender det tilbake til klienten. Typen av Object Result avhenger av om forespørselen til kontrolleren fra klient er valid, og om arbeidet i DAL var vellykket eller ei.

Data Access Layer

Data Access Layer, DAL, har direkte kontakt mot databasen, og kan derfor utføre logiske operasjoner på databasen. Dette kan være så enkelt som å hente ut, legge til, slette eller endre et spesifikt objekt av definert type. Det kan også være mer kompliserte operasjoner, som å filtrere etter objekter definert av en forespørsel. Det er på DAL at den faktiske logikken blir utført på backend.

3.5 Mappestruktur

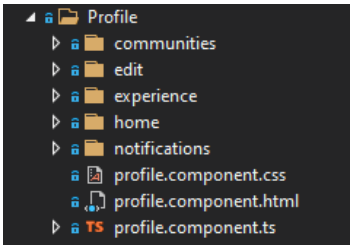
I denne delen av rapporten skal vi se på hvordan vi har strukturert prosjektet, med mapper og underliggende filer.

3.5.1 Frontend

Filer for frontend er hovedsakelig Angular-tjenester, komponenter og HTML-filer som hører til. På frontend ligger det altså logikken som brukeren arbeider opp imot direkte, og hvordan utformingen og utseende er.

App-mappe

I denne mappen ligger de forskjellige delene av tjenesten. Vi har gruppert de forskjellige komponentene, modellen og tjenestene etter bruksområde. Hvis man ser på figur 25, kan man se et eksempel på dette. Her har vi mappen som inneholder det meste av det som brukes innen «Profile»-siden. Denne delen av siden krever flere komponenter, og vi har derfor plassert de innad i sine egne mapper for ordens skyld.



FIGUR 25 - PROFILE-MAPPEN

Node_modules

Her ligger alle pakker som blir brukt i Angular-frontend-prosjektet. Dette er npm-pakker som vi installerer gjennom en Node.js-applikasjon kalt for npm CLI client (Angular, u.å.). Disse pakkene kan være offisielle pakker utgitt av Angular, men som ikke følger med Angular til vanlig. De kan også være pakker utviklet av andre slik som «RxJS», som gir mye funksjonalitet rundt å ha observable objekter og lister (Angular, u.å.).

I package.json-filen kan man se hvilke pakker som er installert, pakkene under «dependencies» er helt nødvendige for å få applikasjonen til å kjøre.

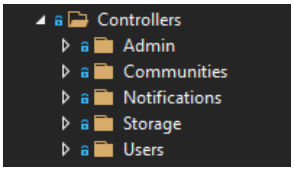
```
{
  "name": "bachelor",
  "version": "0.0.0",
  "scripts": {
    "ng": "ng",
    "start": "ng serve",
    "build": "ng build",
    "build:ssr": "ng run Bachelor:server:dev",
    "test": "ng test",
    "lint": "ng lint",
    "e2e": "ng e2e"
  },
  "private": true,
  "dependencies": {
    "@angular/animations": "8.2.12",
    "@angular/cdk": "^8.2.2",
    "@angular/common": "8.2.12",
    "@angular/compiler": "8.2.12",
    "@angular/core": "8.2.12",
    "@angular/forms": "8.2.12",
    "@angular/material": "^8.2.2",
    "@angular/platform-browser": "8.2.12",
    "@angular/platform-browser-dynamic": "8.2.12",
    "@angular/platform-server": "8.2.12",
    "@angular/router": "8.2.12",
    "@nguniversal/module-map-ngfactory-loader": "8.1.1",
    "aspnet-prerendering": "^3.0.1",
    "bootstrap": "^4.3.1",
    "core-js": "^3.3.3",
    "jquery": "3.4.1",
    "ngx-pipes": "^2.7.5",
    "node-sass": "^4.12.0",
    "oidc-client": "^1.9.1",
    "popper.js": "^1.16.0",
    "protractor": "~5.4.2",
    "rxjs": "^6.5.3",
    "ts-node": "~8.4.1",
    "tslint": "~5.20.0",
    "zone.js": "0.9.1"
  },
```

FIGUR 26 – PACKAGE.JSON

3.5.2 Backend

Controllers

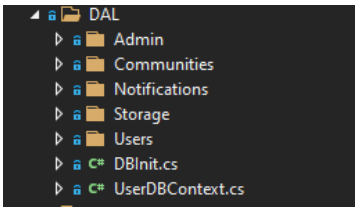
I denne mappen vi har kontrollerene våre som tar imot forespørsler fra frontend.



FIGUR 27 – CONTROLLERS

DAL

Her befinner filer relatert til databasekall og generell databasehåndtering opp med tanke på det laget som heter Data Access Layer.

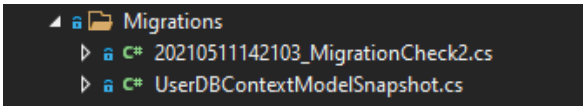


FIGUR 28 - DATA ACCESS LAYER

Migrations

Migrations-mappen inneholder instrukser for hvordan migreringen til Azure-databasen skal foregå. Her ligger det en fil «UserDBContextModelSnapshot», som er den nåværende utformingen av databasemodellen vår. Hvis vi gjør endringer på entiteter som brukes i databasen eller legger til nye, vil denne filen bli oppdatert for hver gang vi legger til en migrasjon.

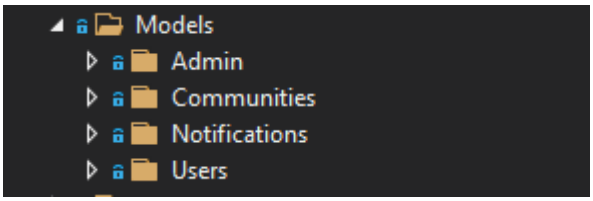
Det ligger også andre filer slik som «MigrationCheck2», som inneholder spesifikke instrukser for den aktuelle migrasjonen. Dette er for at vi enkelt skal kunne oppgradere eller nedgradere databasestrukturen etter behov og endringer.



FIGUR 29 - MIGRATIONS

Models

Her ligger alle klassene vi bruker for prosjektets oppbygning, og det er det som er selve entitetene vi jobber oppimot.



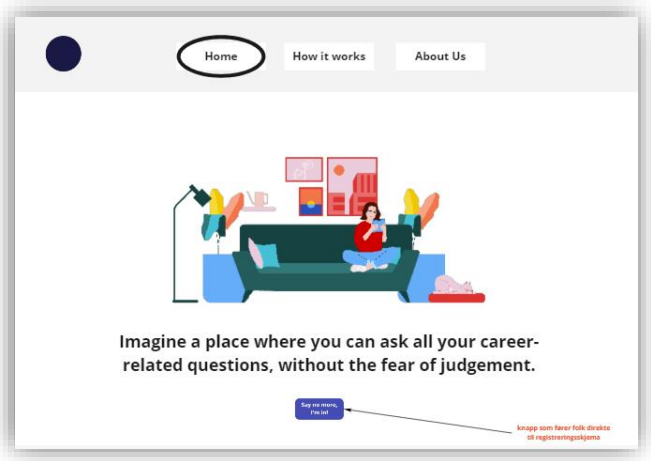
FIGUR 30 - MODELS

3.6 Hoveddeler av programmet

I dette delkapittelet skal vi beskrive hoveddelene av programmet vi har utviklet. Vi kommer først til å forklare kort om landingssiden for mailregistrering, før vi beskriver hoveddelene av webapplikasjonen, KnowONE. Vi kommer til å beskrive de tekniske valgene vi har gjort, samt hvordan disse tekniske valgene fungerer. Deler av programmet som har blitt endret på eller fjernet underveis kommer også til å beskrives.

Side for registrering av mail

Vår første oppgave av bachelorprosjektet gikk ut på å utvikle en informasjonsside, hvor tjenesten skulle beskrives, samt mulighet for å registrere seg for nyhetsbrev. På denne siden har vi lagt stort fokus på design, da dette er brukernes første møte med tjenesten vi senere skulle utvikle. Siden skulle være enkel å bruke, slik at brukerne får den informasjonen vi ønsker å spre gjennom siden.



FIGUR 31 - ØNSKET UTSEENDE AV SIDEN, DELT AV OPPDRAGSGIVER PÅ MIRO

Navbar

For å navigere seg på siden utviklet vi en simpel, men informativ navigasjonsbar, på utviklingsspråket bare kalt navbar, som hele tiden er synlig på toppen av siden. Her har du mulighet til å bytte mellom de to hovedkomponentene, 'hjem' og 'om oss', samt mulighet til å endre språk mellom norsk og engelsk. Navbaren er utviklet med en kombinasjon av egenutviklet HTML og CSS, og Bootstrap. Bootstrap har et enkel design for å utvide navbaren på mindre enheter gjennom en såkalt hamburgermeny, som vil si de tre horisontale linjene over hverandre vist på mobilversjonen i figur 33, men alt av plassering er egenutviklet, da det gir et litt mer personlig preg på utseende.

Hjem

Hjemsiden er det første brukeren møter på siden, og det har derfor vært viktig å finne et sitat som skaper interesse. Siden forklarer ulike deler av tjenesten, og tilbyr brukerne å interagere på siden ved å klikke seg videre, istedenfor bare scrolling, selv om dette også er en mulighet. Er man interessert i å registrere seg med en gang, kan man bli sendt til registreringsdelen ved å trykke på registreringsknappen under teksten øverst på siden. Trenger man litt overbevisning er man nødt til å lese gjennom siden.

FIGUR 32 OG 33 – SAMMENLIGNING MELLOM HJEMSIDEN PÅ DESKTOP (T. V.) OG MOBIL

Om oss

Om oss-siden forklarer hvordan ideen KnowONE kom til og brukerne blir kjent med grunnlegger, Mi Le Hagen. Siden inneholder også informasjon om laget (figur 34), samt oppdragsgivers støttespillere, som har vært til god nytte for oss studenter gjennom prosjektperioden.

FIGUR 34 - AVATARER

Registrering

Delen der man kan registrere mailen sin for å følge nyhetsbrev og oppdatering fra grunnlegger frem mot lansering av MVP'en. Her vil man kunne registrere personlig informasjon, og umiddelbart få en bekreftelsesmail med en hyggelig velkomstmelding fra grunnleggeren.

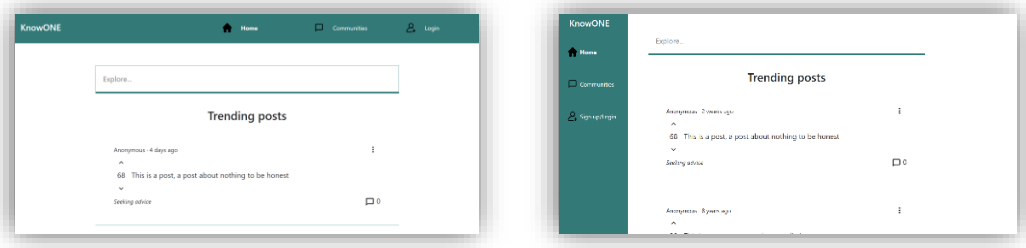
FIGUR 35 - REGISTRERINGSSKJEMA

KnowONE – MVP

Hovedoppgaven vår har vært å utvikle en MVP (Minimum Valuable Product) for ideen til oppdragsgiver. Fokuset har derfor gjennom hele prosjektperioden vært på denne tjenesten. Tjenestens design har mange likheter med registreringssiden vi utviklet i starten av prosjektet, men med tanke på både størrelse av oppgaven og fokus gjennom perioden er denne tjenesten vesentlig større, både designmessig og teknisk. Da registreringssiden krevde lite backend-utvikling, har KnowONE's webapplikasjon balansert mer mellom frontend- og backendutvikling.

Navbar

Navbaren har mange likhetstrekk med den fra mailsiden. Den store forskjellen vises på mindre enheter. I motsetning til på mailsiden flyttes navbaren ned til bunnen av skjermen, for å gi brukeren en form for app-følelse. Ikonene blir værende, mens teksten blir borte da det er relativt mindre plass for tekst på mindre enheter. For å spore hvilken komponent brukeren befinner seg i har vi valgt å bruke det som kalles 'routerLinkActive', som sporer URL. På denne måten vil komponenten brukeren befinner seg i utheves på navbaren. Gjennom store deler av prosjektet har planen vært å plassere navbaren på siden, altså en sidebar. Mot slutten av prosjektet, endret i midlertidig oppdragsgiver mening og vi flyttet den tilbake til toppen, noe vi i gruppen synes var en god beslutning.



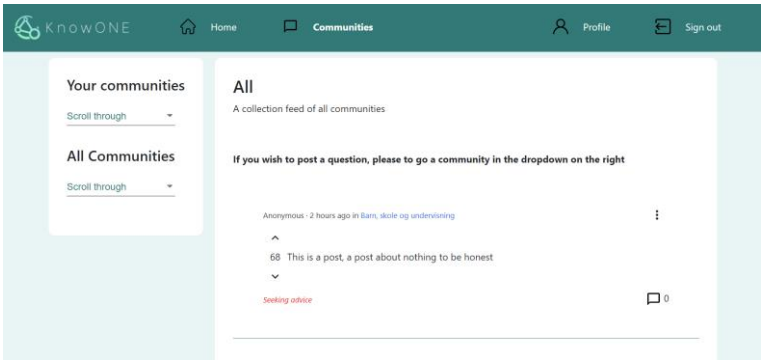
FIGUR 36 OG 37 - SAMMENLIGNING AV NAVBAR PÅ TOPPEN OG SIDEN

Home

Det første som møter brukeren på tjenesten, er home-siden. Her vil brukeren få mulighet til å navigere seg til ulike samfunn ved hjelp av et inputfelt. Brukeren kan også se innlegg som trender på tjenesten, altså innlegg som er mest interagert med. Ved å enten bruke input-feltet, eller trykke på en av de mest populære innleggene, vil brukeren automatisk navigeres til komponenten, 'communities'. Dette blir, som beskrevet i forrige avsnitt, synlig for brukeren på navbaren.

Communities

Communities-komponenten er den delen av tjenesten vi ser for oss at kommer til å bli mest brukt. Her har brukeren mulighet til å navigere seg gjennom de ulike samfunnene og se innlegg, kommentarer og stemmer. Dersom brukeren er innlogget, vil muligheten for å poste innlegg selv, kommentere, stemme eller abonnere/avabonnere på samfunn bli tilgjengelig. For å få denne tilgangen, må man som sagt være innlogget, som kan gjøres i login-komponenten.

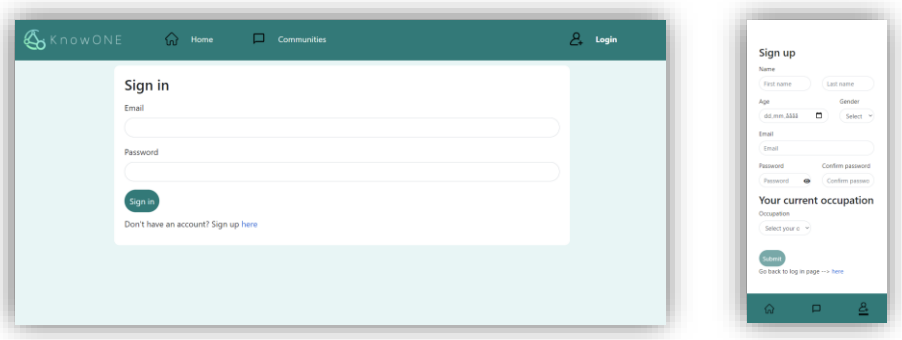


FIGUR 38 - COMMUNITIES

Login

De fleste funksjonene av tjenesten er avhengig av at brukere registrerer profilen sin og er logget inn når de bruker tjenesten. Innloggingsdelen til tjenesten består av to deler: en del hvor man logger inn på en allerede registrert bruker og en del hvor man registrerer en ny bruker. I registreringsdelen, i prosjektet kalt signup-komponenten, blir man bedt om å

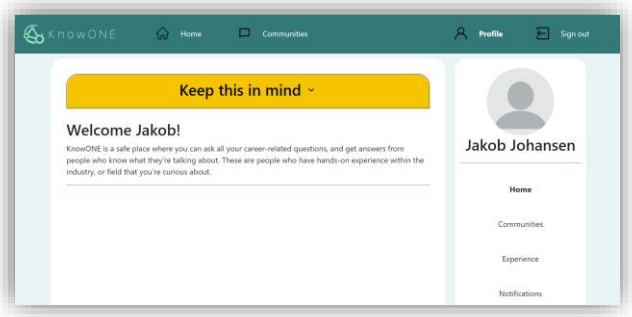
registrere personlig informasjon, passord og nåværende erfaring. Dersom brukerregistreringen er godkjent vil man få valget om å legge til tidligere erfaring. Etter dette er man innlogget automatisk sendt til sin egen profilside, som nå er tilgjengelig i navbaren.



FIGUR 39 OG 40 - SIGN IN PÅ DESKTOP (T. V.) OG SIGN UP PÅ MOBIL

Profile

Per dags dato er det kun mulig å se sin egen profilside. På siden er det mulig å redigere eller legge til erfaringer, se hvilke samfunn man tilhører og se varslinger man mottar i forbindelse med innlegg. Profilsiden er på ingen måter komplett, men ved å redigere sin egen profil vil man kunne presentere seg selv på forskjellige måter, ved for eksempel deling av innlegg.



FIGUR 41 - PROFILSIDEN

3.7 Konklusjon til produktdokumentasjon

Vi har hatt som mål å levere to gode webapplikasjoner, hvor ønsket om videreutvikling skal være enkelt å fylle. Produktene er designet godt både grafisk og teknologisk, slik at det ikke skal være tungt å sette seg inn i verken koden eller programmet i sin helhet. Under arbeidet med prosjektet har vi fått en meget god forståelse på hvorfor god mappestruktur er viktig, og vi mener at det gjenspeiles i prosjektoppbygningen. Vi har fått spisset vår grafiske kunnskap og teknologiske ferdigheter av grafisk utforming ved å bruke nye teknologier for brukerveiledning. Sluttresultatet av disse nye erfaringene og kunnskapene er to webapplikasjoner som både er godt grafisk utformet, men også godt teknologisk designet og skrevet, som vi er stolte av å ha utviklet.

Kapittel 4:

Testdokumentasjon

Innhold

4.1 Hvorfor teste?

Testing av programvare er en essensiell del av utviklingsprosessen. Testing av programvare kan føre til at man finner feil og har muligheten til å rette opp i disse feilene. Testing kan også demonstrere kvaliteten av produktet utviklet, det blir lettere å vise til konkrete resultater og vise at produktet fungerer som ønsket. Vi har fokusert på enhetstesting og brukertester

4.2 Enhetstesting

Enhetstesting går ut på å teste komponenter som kan kjøres individuelt. Enhetstesting er særdeles viktig ettersom dette er det laveste nivået å teste kode på, her er det mulig å oppdage feil ved spesifikke komponenter og dermed sørge for at disse feilene ikke blir overført videre. Med enhetstesting som med all annen testing er det viktig at testene er selvstendige, altså at de kan blir kjørt uten å være avhengige og sammenkoblet med andre deler. Testene skal være raske og dermed lette å kjøre flere ganger (International Software Testing Qualifications Board [ISTQB], 2019, s. 28).

Vi enhetstestet kontrollerne våre, her var poenget å se at vi fikk riktige data tilbake. Denne dataen kan enten være et objekt eller en samling av objekter utifra hva forespørselen er og hva dataen skal brukes til. Noen ganger finnes ikke dataen vi spør om, eller forespørselen ble gjort på ugyldig format, da vil man få tilbake en HTTP-melding som gjenspeiler dette. Begge tilfellene med «brukbar» data og feilmelding er gyldige hendelser i løpet av et program, vi måtte derfor teste får begge utfall.

Ut fra våre kontrollere har de fleste en del felles utfall:

<u>OK</u>

Hvis forespørselen er gjort på riktig format, har autentisering hvis det er nødvendig, og det blir funnet data som kan returneres.

NoAuth

Står for «No Authentication», og gjenspeiler at man gjør en forespørsel til kontrolleren uten å ha en gyldig «Authorization» verdi i HTTP headeren. Ved noen kall som for eksempel å hente ut data om en bruker, så kreves det at man har en token fra JWT som inneholder samme bruker ID som den brukeren man forsøker å hente data om.

ModelState

Hvis det blir sendt inn hele objekter, som ved en opprettelse av en kommentar eller innlegg via POST forespørsel, blir dette objektet testet for inputvalidering. Hvis den ikke stemmer med våre krav, blir det avslått.

NotFound

Denne feilen oppstår hvis man gjør en forespørsel om en entitet, men denne entiteten ikke blir funnet i databasen vår.

4.2.1 Oppsett av tester

Mock og arrange, act, assert

Ved oppsett av enhetstester er det tre primære hendelser. Vi må «arrange» dataen og oppsettet for testene. Vi må «act» som er å utføre selve oppgaven som ønsker å bli testet, og til slutt «assert» for å se om retur dataen er slik vi forventet at den skulle være.

I våre tester bruker vi en metode som kalles «mocking», dette betyr at vi forfalsker en del metoder. Dette ettersom vi ikke har lyst til å jobbe direkte oppimot database laget som ligger under kontrollerene våre. Ved å mocke repositoriet i det underliggende laget kan vi også spesifisere hvilke data vi vil at metodene skal returnere. Vi kan derfor sikre oss at utfallet er som forventet og se at metoden oppfører seg riktig utifra hvilken data den får tilbake (Kayal, 2018).

```
[Fact]
0 references
public async Task PostCommentOK()
{
    //Arrange
    DateTime now = DateTime.Now;
    Community com1 = new Community { Id = 1, Title = "Community 1", Description = "Test community", Level = 0, Communities = new List<Community>() };
    User user = new User { Id = 3, Age = now, Email = "magnushjk@gmail.com", Firstname = "Magnus", Lastname = "Kristiansen", Gender = "Man", Role = "admin",
    Post post = new Post { Id = 1, Community = com1, Comment = new List<Comment>(), Anonymous = true, Date = "2021-01-01T12:00:00", Downvotes = 0, Upvotes =
    Comment comment = new Comment { Id = 1, Post = post, User = user, Text = "Test comment", Date = "2021-01-01T12:00:00", Upvotes = 0, Downvotes = 0, Respor

    var httpContext = new DefaultHttpContext();
    httpContext.Request.Headers["Authorization"] = _Auth;


    mockRep.Setup(s => s.PostComment(1, comment)).ReturnsAsync(true);
    mockJWT.Setup(j => j.ValidateWithAccess(httpContext, user.Id)).Returns(true);

    var CommentController = new CommentController(mockRep.Object, mockJWT.Object)
    {
        ControllerContext = new ControllerContext()
        {
            HttpContext = httpContext
        }
    };

    //Act
    var resultat = await CommentController.PostComment(1, comment) as OkObjectResult;

    //Assert
    Assert.IsType<OkObjectResult>(resultat);
    Assert.Equal((int)HttpStatusCode.OK, resultat.StatusCode);
    Assert.Equal(true, resultat.Value);

}
```

FIGUR 42 - TEST AV Å LEGGE TIL KOMMENTAR

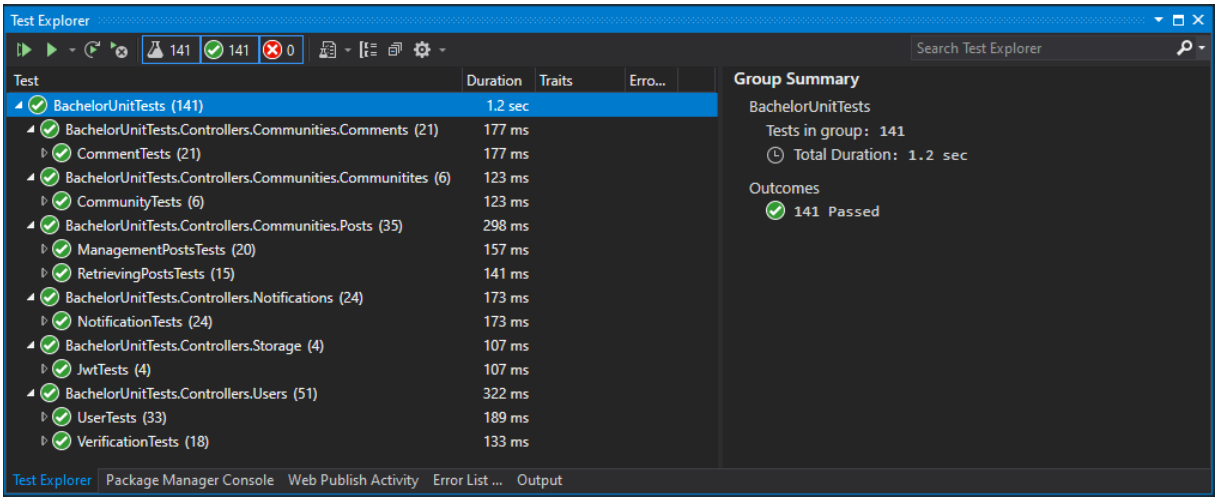
I figur 42 ser vi en test av å legge til en kommentar. Først må vi sette opp all data som skal sendes. Her har vi en kommentar med alle nødvendige objekter som skal være tilknyttet en kommentar, som en post, community og bruker.

Så er vi nødt til å sette HTTP header verdien «Authorization» til en gyldig verdi, ettersom denne forespørselen krever autentisering.

Deretter blir mock satt opp, vi er nødt til å mocke to repositorier. Den første som utfører arbeid på selve dataen, og en som fanger opp HTTP konteksten og sjekker at autentiseringen er valid. Når vi mocker kan vi spesifisere hvilket svar vi vil at metoden skal returnere. Ettersom dette er et gyldig tilfelle, skal de begge returnere «true».

Når dette er satt opp, kan vi utføre forespørselen og fange det opp i en variabel. På variabelen kan vi da sjekke om den er lik det vi forventet å få tilbake ved å sammenligne og sjekke at den er av riktig datatype.

4.2.2 Resultater



FIGUR 43 - TESTRESULTATER

I figur 43 kan man se resultatene etter at testene har blitt gjennomført.

4.2.3 Konklusjon til enhetstesting

Med tanke på testing er det viktig å reflektere over de syv testprinsippene, som for eksempel «Testing viser at feil er til stede, ikke deres fravær» (International Software Testing Qualifications Board [ISTQB], 2019, s. 15). Det er derfor alltid mulig å teste mer og få større sikkerhet i kvaliteten av produktet. Vi kunne ha videreført testingen til høyere nivåer som, integrasjonstesting og systemtesting. Vi valgte å fokusere på enhetstester for å sikre kvaliteten på enkeltstående metoder. Dette gjorde at vi oppdaget en del små feil. Samtidig fikk vi endret på en del retur verdier slik at å jobbe oppimot metodene fulgte mest mulig lik logikk over hele prosjektet.

4.3 Brukertesting

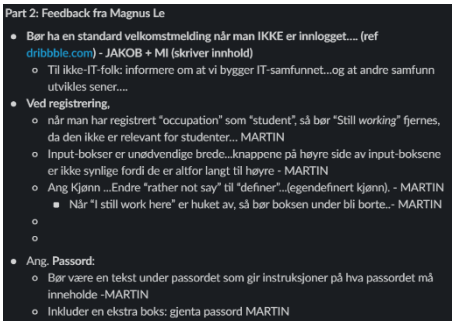
4.3.1 Innledning

For å finne ut om et brukergrensesnitt fungerer godt, er man avhengige av å utføre brukertester. Brukertestene gjennomføres på personer som ikke har tilknytting eller erfaring med tjenesten. Vi som har utviklet tjenesten vet allerede hvordan den fungerer og hva som

kan gjøres. Det vil derfor være meningsløst om vi skal teste vårt eget produkt, så vi ønsket å la nøytrale brukere, uten kunnskap om hvordan tjenesten fungerer, teste den. Før man skal gjennomføre brukertester, kan det være greit å ha en testplan. Hva er hensikten med testen? Hvorfor ønsker vi å teste?

Brukertestene ble gjennomført over kommunikasjonstjenesten Discord, da denne tjenesten tilbyr et godt opplegg for videodeling, slik at vi enkelt kan både se og høre hva brukerne tenker når de tester tjenesten vår.

Brukertestingen ble gjennomført mot slutten av prosjektperioden. Planen var egentlig å få på plass brukertesting i god tid før slutten, men på grunn av mangler i funksjonalitet ble brukertestingen forskjøvet. Selv om vi ikke fikk testet med nøytrale brukere før mot slutten, har vi hatt jevnlig gjennomgang av brukergrensenittet med vår mentor, Magnus Le. Han har gjennom hele prosjektperioden gitt tilbakemeldinger på brukergrensesnittet og kommet med råd og tips.


Part 2: Feedback fra Magnus Le

- **Bør ha en standard velkomstmelding når man IKKE er innlogget.... (ref dribbble.com) - JAKOB + MI (skriver innhold)**
  - Til ikke-IT-folk: informere om at vi bygger IT-samfunnet...og at andre samfunn utvikles sener....
- **Ved registrering,**
  - når man har registrert "occupation" som "student", så bør "Still *working*" fjernes, da den ikke er relevant for studenter... MARTIN
  - Input-bokser er unødvendige brede...knappene på høyre side av input-boksene er ikke synlige fordi de er altfor langt til høyre - MARTIN
  - Ang Kjønn ...Endre "rather not say" til "definer"...(egendefinert kjønn). - MARTIN
    - Når "I still work here" er huket av, så bør boksen under bli borte..- MARTIN
  -
  -
- Ang. **Passord:**
  - Bør være en tekst under passordet som gir instruksjoner på hva passordet må inneholde -MARTIN
  - Inkluder en ekstra boks: gjenta passord MARTIN

FIGUR 44 - UTDRAG AV TILBAKEMELDING FRA MAGNUS LE

4.3.2 Testplan

Vi ønsket å brukerteste KnowONE's MVP som vi har utviklet. Hensikten med brukertestene våre var i hovedsak å finne ut om tjenesten vi har utviklet gir mening, og om det er noe helt vanlige mennesker ville brukt i sitt daglige liv. Vi ble derfor enige om å gjennomføre tester over internett, på grunn av Covid-19, og vi skulle gjennomføre et par tester hver. Vi ønsket å gi brukere forskjellige oppgaver å utføre, og deretter stille spørsmål.

Oppgavene var som følger:

- Registrer deg
- Legg til erfaring på profilsiden
- Abonner til et samfunn
- Post et innlegg

- Svar på et innlegg

Dette er oppgaver som for oss utviklere er relativt simple, men hvor vanskelig er det egentlig å utføre slike oppgaver på et brukergrensesnitt som er helt nytt, dersom man ikke har forkunnskaper om hvordan tjenesten skal fungere?

Når oppgavene er gjennomførte skal vi stille enkle spørsmål, for å få et innblikk i hvordan brukere opplever tjenesten vår. Dette er spørsmål som beskriver hva brukerne synes om tjenesten, om de forsto tjenesten og om det er noe som kunne vært gjort bedre. Denne dataen ville vi samle inn gjennom en spørreundersøkelse.

4.3.3 Testrapport

Formålet med brukertestingen var å få innblikk i hvordan nye brukere opplever tjenesten, om de skjønner hvordan den skal brukes og om det er en tjeneste de kunne tenke seg å bruke, dersom den hadde vært tilgjengelig.

Siden testingen skulle foregå på hovedprosjektet vårt, altså KnowONE's MVP, ble hver bruker tildelt en lenke hvor tjenesten kunne testes, samtidig som at de ble bedt om å dele skjermen sin. Grunnen til at vi ønsket å se skjermen, var for å se hvordan de navigerte seg for å utføre oppgavene sine.

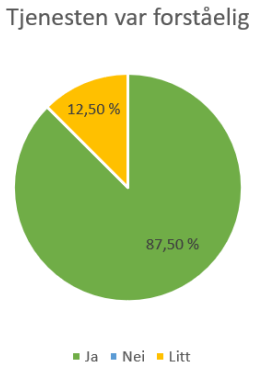
Som beskrevet i testplanen, ble brukerne bedt om å utføre spesifikke oppgaver. Vi la merke til at det for brukerne virket som at det var opplagt hvor man skulle trykke for å utføre hver oppgave. På grunn av litt tekniske komplikasjoner ble dessverre ikke brukertestingen optimal, da enkelte funksjonaliteter ikke fungerte optimalt. Likevel klarte alle brukerne å utføre oppgavene de fikk tildelt.

Med tanke på at brukertestingen ble gjennomført såpass sent i prosjektperioden, var det begrenset hva vi kunne gjøre av endringer ut fra tilbakemeldingene. Vi hadde i midlertidig tid til å utføre noen endringer som ble etterspurt av brukerne, blant annet oppsette av navbaren, bakgrunnsfarger og at ved logoklikk blir man sendt tilbake til «Home».

4.3.4 Resultater

Som skrevet i testplanen skulle alle deltakere gjennomføre en spørreundersøkelse, for å kartlegge hva en vanlig bruker kommer til å tenke om tjenesten. Vi ønsket å undersøke om tjenesten var forståelig og hvilket inntrykk brukerne fikk av tjenesten.

Første spørsmål på undersøkelsen var om brukerne synes tjenesten var forståelig. Dette er et viktig spørsmål, da tjenesten er avhengig av å være forståelig for at brukere skal ha et ønske om å bruke tjenesten. Spørsmålet hadde tre svaralternativer, «Ja», «Nei» og «Litt». Resultatet (figur 45) viser at over 80% av tjenesten synes tjenesten var forståelig, mens de resterende brukerne synes tjenesten var litt forståelig. Videre ønsket vi tilbakemelding på om noe på tjenesten var misledende eller vanskelig å forstå. I figur 46 ser vi et utdrag av tilbakemeldingene, hvor vi kan se at noen av brukerne ønsket en beskrivelse av tjenesten før man registrerer seg.

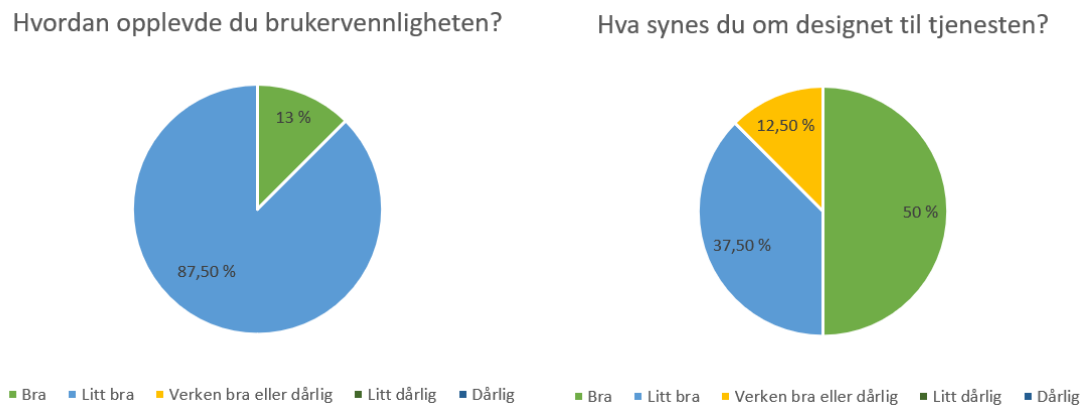


FIGUR 45 - FØRSTE SPØRSMÅL

Mangler en forside som beskriver tjenesten med noen få ord.
Man vet ikke helt hva siden gjør før man har laget en bruker. Er først i din egen profil du får vite hva KnowOne er.
Vil gjerne ha notifikasjon fjernet etter jeg har trykket på den.
Hvordan legge til nytt innlegg

FIGUR 46 - ER DET NOE SOM VAR MISLEDENDE / VANSKELIG Å FORSTÅ?

Videre ønsket vi tilbakemelding på tjenestens brukervennlighet og design, noe vi fikk gode tilbakemeldinger på. Nesten 90% av testerne svarte at de synes tjenesten var «Litt bra», som er den nest høyeste scoren i undersøkelsen, mens resterende svarte at brukervennligheten var «Bra». På spørsmål om design svarte 37,5% at designet var «Litt bra», halvparten av testerne svarte at designet var «Bra», den høyeste scoren. Resten svarte at designet verken var bra eller dårlig, altså nøytralt. Resultatene, vist i figur 47 og figur 48, viser at tjenestens design og brukervennligheten er bra, og med tanke på at dette er en MVP, mener vi at tjenesten er et godt utgangspunkt for oppdragsgiver.



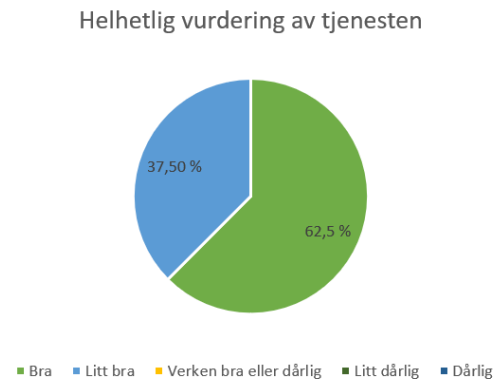
FIGUR 47 OG 48 - RESULTATER ANGÅENDE BRUKERVENNLIGHET (T. V.) OG DESIGN

På spørsmål om hva brukerne synes var spesielt bra med tjenesten svarte 50% at ideen og konseptet var bra. Dette virker lovende med tanke på oppdragsgivers mål med tjenesten. Testerne fikk også spørsmål om hva som må til for at de skulle brukt KnowONE til vanlig, og flesteparten svarte at de ville brukt tjenesten dersom de var på utkikk etter svar, enten i form av karriereveiledning eller generelt. Èn bruker svarte at det var usikkert hvordan tjenesten kunne passe testerens behov. Oppdragsgiver får dermed en indikasjon på at tjenesten bør tilpasses mennesker som ønsker råd og svar innen diverse felt, noe som har vært oppdragsgivers hovedmålgruppe.

At det er mange brukere som melder seg på
Spørsmål om studier/karriere
Være på jakt etter studie eller lete fram svar på noe innenfor mitt eget felt.
Vanskelig å si, ser ikke helt hvordan tjenesten passer mitt behov
At andre folk som kan svare på spørsmål bruker det

FIGUR 49 - HVA MÅ TIL FOR AT DU SKAL BRUKE KNOWONE?

På spørsmål om helhetlig vurdering fikk vi også gode tilbakemeldinger. Figur 50 viser at hele 62,5% av testerne synes tjenesten var bra, mens de resterende 37,5% synes tjenesten var litt bra.



FIGUR 50 - HELHETLIG VURDERING AV TJENESTEN

4.3.5 Konklusjon til brukertester

Etter gjennomførte brukertester, har både oppdragsgiver og prosjektgruppen fått gode tilbakemeldinger, både i form av hva vi har gjort bra og hva som kunne vært gjort bedre. I denne testen ønsket vi hovedsakelig å finne ut om tjenesten vi har utviklet gir mening for en nøytral bruker og om personer ville brukt en slik tjeneste. Vi er meget fornøyde med tilbakemeldingene, og ut fra disse tilbakemeldingene mener vi at vi kan overrekke oppdragsgiver et produkt med godt utgangspunkt i forhold til oppdragsgivers mål og visjon, og gode muligheter for videreutvikling.

Kapittel 5:

Referanseliste og vedlegg

5.1 Referanseliste

Angular. (u.å.). Angular. Hentet 11. mai 2021 fra https://angular.io/

Angular Material. (u.å.). Angular Material. Hentet 25.05.2021 fra https://material.angular.io/.

Bootstrap. (u.å.). Bootstrap. Hentet 25.05.2021 fra https://getbootstrap.com/.

Code Project. (2008, 8 april). Simple Example of MVC (Model View Controller) Design Pattern for Abstraction. https://www.codeproject.com/Articles/25057/Simple-Example-of-MVC-Model-View-Controller-Design

Foss-Pedersen, R. J. (2017, 24. august). Fem tips til deg som skal gjennomføre brukertester. *UX-bloggen*. https://www.usit.uio.no/om/organisasjon/bnt/web/ux/blogg/2017/brukertest.html?vrtx=tags

Hassan, Z.U (Sist oppdatert 2020, 03 desember). Action Result In ASP.NET MVC. C# Corner. https://www.c-sharpcorner.com/article/action-result-in-asp-net-mvc/

International Software Testing Qualifications Board. (2019). Sertifisert Tester: Pensum for grunnivå. http://www.istqb-norge.no/wp/wp-content/uploads/2019/04/CTFL-2018-Pensum-norsk-v1.0.pdf

JWT. (u.å.). JWT. Hentet 21. mai 2021 fra https://jwt.io/

Kayal, S. (Sist oppdatert 2018, 25 oktober). Fundamentals of Unit Testing: Understand Mock Object in Unit Testing. C# Corner. https://www.c-sharpcorner.com/UploadFile/dacca2/fundamental-of-unit-testing-understand-mock-object-in-unit/

Shifatul Islam. (2018, 8. oktober). *Angular – date ago pipe (minutes / hours / days / months / years ago)*. Medium. https://medium.com/@thunderroid/angular-date-ago-pipe-minutes-hours-days-months-years-ago-c4b5efae5fe5.

MuleSoft. (u.å.). *What is an API?*. MuleSoft. https://www.mulesoft.com/resources/api/what-is-an-api

RestfulAPI. (u.å.). *What is REST*. Restful API. https://restfulapi.net/

Sommerville, I. (2015). *Software Engineering*. (10.utg.). Pearson.

Sandnes, F., E. (2018). *Universell utforming av IKT-systemer.* (2. utg.). Universitetsforlaget.

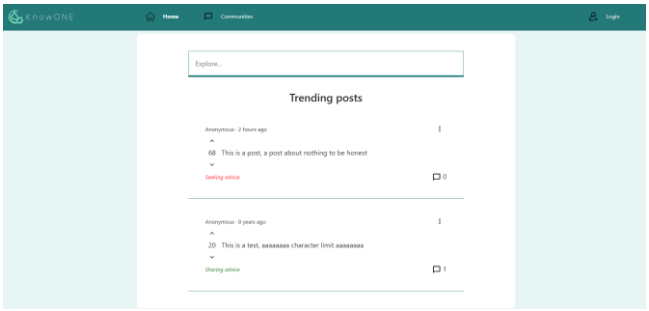
Vaniukov, S. (u. å.). Colors in UI Design: A Guide for Creating the Perfect UI. *Usability Geek.* https://usabilitygeek.com/colors-in-ui-design-a-guide-for-creating-the-perfect-ui/

5.2 Vedlegg

Mailregistreringssiden

FIGUR 51 - KNOWONE.NO

Hovedprosjetktet, KnowONE MVP



FIGUR 52 - KNOWONE MVP

Pipe hentet fra internett

For å vise hvor lenge siden diverse innlegg på hovedprosjektet ble lagt ut, har vi hentet en pipe fra en Medium-artikkel, skrevet av Shifatul Islam. Koden hentet fra denne siden er lagt til frontendmappen 'pipes', i prosjekt vårt (2018, Shifatul Islam).

5.3 Begrepsliste

Angular

- Angular er et utviklingsrammeverk for webapplikasjoner (Angular, u.å.).

API

- API er et akronym for 'Application Programming Interface'. Et API lar to applikasjoner dele data mellom hverandre, for eksempel frontend og backend (https://www.mulesoft.com/resources/api/what-is-an-api)

Asynkront

- Asynkront brukes i webapplikasjoner dersom utviklerne ønsker at en prosess skal kjøres samtidig som flyten i programmet fortsetter som vanlig.

Autentisering

- Autentisering er et viktig sikkerhetsprinsipp som handler om hvordan en bruker får sin identitet bekreftet

Autorisering

- Autorisering er et sikkerhetsprinsipp som handler om hvordan noen kan få tilgang til en spesifikk ressurs.

Backend

- Backend omtales den delen av koden som utfører operasjoner med databasen og arrangerer server-struktur.

CLI

- ClI, eller Command Line Interface, brukes til å kjøre funksjoner, i vårt tilfelle brukte vi det til å installer NPM pakker.

Database

- Database er en samling av data

Dependencies

- Dependencies, eller avhengigheter, brukes rundt IT faget til å forklare at en type software eller pakke er avhengig av en annen pakke for å fungere normalt.

Frontend

- Frontend omtales den delen av koden som designer nettsiden og tar imot informasjon fra brukere.

Git

- Git er et versjonskontroll-system som tillater brukere å legge prosjektet sitt på et sted som kan hentes ut eller oppdateres til nye brukere.

GitHub

- GitHub er en platform som lar brukere laste opp prosjektet sitt og samhandle med det via GIT

IDE

- En IDE, eller Integrated Development Environment, er et program som kan brukes til å utvikle systemer eller applikasjoner via programmeringsspråk.

JSON

- JSON, eller JavaScript Object Notation er et skriftformat laget for transportering av data.

Kravspesifikasjon

- Kravspesifikasjon er en beskrivelse av egenskaper produktet skal inneholde.

Microsoft Azure

- Microsoft Azure er en plattform plassert i skyen som tilbyr for eksempel muligheten til å lansere din nettside på en server.

NPM

- NPM, eller Node Package Manager, er et bibliotek for kode pakker.

REST

- REST er et akronym for 'Representational State Transfer' og brukes ofte rundt arbeid med API. REST er enkelt oppsummert en arkitektur for hvordan data blir overført (https://restfulapi.net/).