UNIVERSITETET I AGDER

Forside

IS-304: 2024

Tittel: Effektivisering av brukerstøtte for Felles studentsystem ved bruk av kunstig intelligens

Emnekode	IS-304
Emnenavn	Bacheloroppgave i informasjonssystemer
Emneansvarlig:	Hallgeir Nilsen & Geir Inge Hausvik
Veileder	Janis Gailis
Oppdragsgiver:	Kevin Benjamin Zeppo Adriaansen (UiA)

Studenter:

Etternavn	**Fornavn**
Falkum	Trym
Grajcevci	Edi
Meen	Isak Kvernes
Nguyen	Christian Thien Truong Luong
Vu	Huy Trong

Jeg/vi bekrefter at vi ikke siterer eller på annen måte bruker andres arbeider uten at dette er oppgitt, og at alle referanser er oppgitt i litteraturlisten.	**JA**
Kan besvarelsen brukes til undervisningsformål?	**JA**
Vi bekrefter at alle i gruppa har bidratt til besvarelsen	**JA**

Forord

Bacheloroppgaven er en del av emnet IS-304, hvor hovedformålet er å gi studentene mulighet til å praktisere kunnskaper og ferdigheter vi har anskaffet oss gjennom bachelorstudiet, ved å fordype oss i et prosjekt relatert til et informasjonssystem i en reell setting.

Vi ønsker å rette en hjertelig takk til alle som har bidratt til utviklingen av denne bacheloroppgaven.

Først og fremst vil vi takke professor Janis Gailis for hans uvurderlige veiledning og støtte gjennom hele prosjektet. Hans ekspertise, innsikt og interesse i prosjektet har vært avgjørende for å veilede oss gjennom de ulike fasene av forskningen og skrivingen av rapporten.

Vi vil også takke Kevin Benjamin Zeppo Adriaansen for hans rolle som oppdragsgiver, mentor og forbindelsesledd til andre ressurser og personer som har vært involvert i prosjektet. Hans engasjement har vært til stor hjelp i å forme vår tilnærming og løsninger. En spesiell takk rettes også til Mauricio Omar Cifuentes fra IT-avdelingen for sin verdifulle hjelp med utstyr og teknisk støtte.

Sammendrag

Denne rapporten dokumenterer et konseptutprøvingsprosjekt utført i samarbeid med UiA. Målet med prosjektet er å basere kunstig intelligens på egne data til å besvare saker automatisk, eller som en assistent. Analysen av Felles studentsystem viste behovet for bedre brukerstøtte grunnet et utdatert brukergrensesnitt som førte til brukervansker og feil. Prosjektet ble derfor rettet mot å redusere ventetiden og antall henvendelser til FS-Hjelp. Etter gjennomføring av prosjektet har gruppen fått verdifullt læringsutbytte og erfaring innen prosjektstyring og systemutvikling.

Innholdsfortegnelse

Liste over figurer

Liste over tabeller

1. Introduksjon

Bacheloren for IT og Informasjonssystemer på Universitetet i Agder (UiA) innebærer en samarbeidspartner. I dette prosjektet inngikk gruppen et samarbeid med administrasjonen på UiA vårsemesteret 2024, hvor prosjektet var en konseptutprøving av å effektivisere brukerstøtte ved hjelp av kunstig intelligens.

Rapporten er satt opp på følgende måte:

"Analyse": Introduserer problemdomenet.
"Prosjektstyring og gjennomføring": Starter med prosjektstart (Pre-sprint), etterfulgt av åtte faser (Sprint 1-8) av prosjektets gjennomføring, som er basert på Scrum.
"Evaluering": En evaluering av prosjektet

Etter "Analyse" følger rapporten en kronologisk struktur. Dette for å dokumentere prosjektets prosess, samt muliggjøre etterprøving av konseptet. Siden rapporten er satt opp slik som den er, blir hver sprint kort beskrevet i starten av hvert Sprint-kapittel, hvor man også vil finne en punktliste over "Verktøy og metoder". En fullstendig liste over disse, inkludert sidetall hvor verktøyet eller metoden først blir presentert, er vedlagt i vedlegg 3. Det er også lagt ved en systemarkitektur i den første sprinten, samt en oppdatert versjon i slutten av de sprintene denne forandres. Dette vil hjelpe leseren med å sette seg inn i gruppens tankegang på det daværende tidspunktet i prosjektet.

I sin helhet er rapporten først og fremst utformet for emneansvarlige og sensor. Ytterligere er rapportens dokumentasjon av gruppens forskning og tekniske gjennomføring verdifull for prosjektets interessenter, spesielt oppdragsgiver og IT avdelingen ved UiA, til eventuell videreutvikling og vedlikehold. For disse aktørene er analysen, delkapitlene i hver av sprintene som heter "Sprint gjennomføring", og konklusjonskapitlet de fremstående og anbefalte delene av prosjektrapporten.

1.1 Bakgrunn

I november 2023 inngikk gruppen et samarbeid med en annen bedrift om en bacheloroppgave innen informasjonssikkerhet. I den påfølgende kommunikasjonen med bedriften oppstod det dessverre flere utfordringer. Først og fremst indikerte bedriften at de ikke kunne garantere tilstrekkelig støtte eller oppfølging gjennom semesteret grunnet begrensede ressurser, som ble bekreftet nær nyttår. Utover dette kunne ikke bedriften tilby en klar problemstilling for bacheloroppgaven, og det ble opp til gruppen å definere prosjektets omfang og formål selv. Disse utfordringene ble avgjørende for beslutningen om å ikke gå videre med samarbeidet, da gruppen mente at kontinuerlig kommunikasjon med oppdragsgiver gjennom semesteret, samt et reelt problem fra bedriften, var essensielt for et velfungerende bachelor-samarbeid.

Ved overgangen til 2024 tok gruppen derfor kontakt med emneansvarlig Hallgeir Nilsen for å forklare utfordringen. Etter flere dialoger med Hallgeir, og et par møter med andre bedrifter, ble gruppen henvist til Kevin Adriaansen fra UiA Administrasjonen. Han presenterte en potensiell problemstilling som var svært relevant og falt innenfor interessefeltet til gruppemedlemmene. Etter en intern diskusjon ble beslutningen tatt om å gå videre med samarbeidet med UiA, og bachelorløpet ble startet med et forsinket utgangspunkt.

2. Analyse

Gruppen har blitt tildelt et bachelorprosjekt hvor oppgaven var å bruke kunstig intelligens til å effektivisere brukerstøtten til Felles studentsystem (FS) ved Universitetet i Agder. FS er et studentinformasjonssystem (SIS) som består av databaser, integrasjoner og brukerapplikasjoner (Felles studentsystem, u.å), og er kjernesystemet av all studentrelatert informasjon og studiedata for nesten alle høyere utdanningsinstitusjoner i Norge. Systemet ble etablert i 1995 og har inntil nå blitt kontinuerlig modernisert.

Basert på en nylig evaluering utført av Gartner Group, har det blitt fastslått at den underliggende tekniske arkitekturen til FS forblir robust nok, men at det eksisterende Frontend-designet ikke lengre tilfredsstiller moderne standarder og måtte erstattes fullstendig. Med eksempler der lav brukervennlighet og komplekst brukergrensesnitt har medført kritiske feil med konsekvenser for institusjonene og for studentene (Lehne & Nisar, 2020, s. 12).

Det er et økende behov for støtte fra FS-Hjelp på Universitetet i Agder, da den nåværende situasjonen fører til at brukere opplever vanskeligheter med å orientere seg, som beskrevet av Gartner Group over. Vårt prosjekt tar sikte på å utforske potensialet for effektivisering av brukerstøtte ved å implementere kunstig intelligens. Ved bruk av AI sikter vi mot å redusere ventetiden for brukerstøtte og mengden klassiske henvendelser, samt frigjøre verdifulle arbeidstimer for rådgiverne i FS-Hjelp, slik at de kan fokusere på mer komplekse oppgaver.

Kunstig intelligens, også kjent som AI, er en paraplybetegnelse for teknologier med evnen til å simulere menneskelig intelligens. Dette inkluderer evnen til å lære, tolke kompleks informasjon, utføre analyser, gjenkjenne mønstre, og ta avgjørelser (IBM, u.å.). Som nevnt over så er kjernen i prosjektet fokusert på anvendelsen av kunstig intelligens. Denne teknologien er avgjørende for prosjektets mål om å skape mer effektiv og intuitiv brukerstøtte for FS, og har gitt oss muligheten til å automatisere interaksjoner for å gi brukerne rask og relevant assistanse.

Prosjektbeskrivelsen utarbeidet av oppdragsgiver omfattet en rekke funksjoner og mål som var grunnlaget for tilnærmingen til gruppen. En av de mest sentrale funksjonene som

løsningen måtte ha var muligheten til å designe eller laste inn informasjon til AI-assistenten, som den skulle basere sin kunnskap på. Informasjonen skulle også kunne enkelt bli byttet ut og oppdatert, samt være mulig å supplere eller fjerne tidligere kunnskap. Dette innebærte at FS-Hjelp skulle kunne strukturere og tilrettelegge kunnskapen som AI-assistenten opererte med, slik at den kunne gi relevante og nøyaktige svar på brukernes henvendelser, ettersom FS-veiledningene ikke lå offentlig på nett for bruk av standard AI.

Videre måtte løsningen kunne laste- eller hente inn mottatte FS-supportsaker og gi forslag til besvarelse på henvendelsen. I tilfeller der spørsmålet ikke kunne besvares direkte, men krevde handling, skulle AI-assistenten foreslå prosedyren eller kunnskapsartiklene brukeren kunne benytte. AI-assistenten måtte også kunne identifisere og spesifisere eventuelle problemer den opplevde med henvisninger eller deler av den, for å tydeliggjøre hvor det var potensial for forbedringer i brukerveiledningene. Samtidig måtte gruppen ivareta behov fra interessentene under utvikling av løsningen. En oversikt over gruppens interessenter er vedlagt under, i tabell 1.

Interresseanalyse		
Interresent (Rolle)	Navn	Behov som må ivaretas
Oppdragsgiver	Universitetet i Agder	Vite at gruppen har et tilfredstillende og godt læringsutbytte, samt følger UiA sine rettningslinjer
Produkteier (oppdragsgiver)	Kevin Benjamin Zeppo Adriaansen (UiA)	Kontroll over prosjektetes framdrift og fremtidige planer
Sluttbruker	FS brukere & rådgivere i fellesadministrasjon på UiA	Generelle ønsker, og interesse for bruk
Veileder	Janis Gailis	Status, framdrift og metodikk i prosjektet
Emneansvarlig	Hallgeir Nilsen & Geir Inge hausvik	Vite om framdrift i prosjekt og opprettholdt kommunikasjon med bedrift
Ressursleverandør	Mauricio Omar Cifuentes (IT-avdelingen på UiA)	Informert om behovet for verktøy, maskinvare eller økonomisk bistand

Tabell 1: Interessentanalyse

Oppdragsgiver hadde også identifisert potensielle funksjoner for videre utvikling av løsningen, deriblant håndtering av ulike “fagfelter” innenfor systemet, som eksamen, opptak og tilrettelegging. I tillegg var det et behov for å kunne ta imot spesifikke domener som UiA.no og UiO.no for å utvide kunnskapsbasen til AI-assistenten.

Målet med prosjektet var fundamentalt konseptutprøving, for å potensielt forbedre eksisterende brukerveiledninger ved bruk av kunstig intelligens. Kravene fra oppdragsgiver inkluderte at studentene “skal gi opplæring i generelt vedlikehold av løsningen” og “presentere oppgaven for UiA sine interessenter i et separat møte.” Noe som vil ta plass mellom rapportens leveranse og muntlig eksamen. Det skulle også holdes statusmøter en gang i uken gjennom prosjektets varighet, som arrangeres og ledes av studentene.

Det er viktig å understreke at implementeringen av AI-assistenten for å forbedre brukerstøtten er først og fremst ment som en midlertidig løsning. Innsatsen er rettet mot å minske de umiddelbare utfordringene brukerne opplever, inntil et nytt og mer brukervennlig grensesnitt for FS er utviklet og implementert. På den måten fungerer AI-løsningen som en bro mellom den eksisterende utfordrende brukeropplevelsen og det kommende brukergrensesnittet. Og etter implementeringen av det nye brukergrensesnittet, kan

løsningen fortsette å være et verdifullt verktøy for å støtte brukerne, forhåpentligvis med redusert behov for støtte.

3. Prosjektstyring og gjennomføring

Prosjektstyring handler om å anvende kunnskap, metoder eller verktøy for å sikre at kravspesifikasjonene blir tilfredsstilt. Dette hjelper gruppen å sette forventninger, definere mål, beholde kvalitet, redusere risiko og overse fremgang i prosjektet (Aston, 2024). Denne delen av rapporten skal fremvise ulike deler av prosjektstyring som gruppen har benyttet, i tillegg til prosjektstarten og gjennomføringen av prosjektet.

Sprint	Periode	Hovedmål	Hovedaktiviteter
1	23.01 - 16.02	Forståelse av systemet og domene	Tankemodell, systemarkitektur, kunnskapsdatabase, risikovurdering
2	19.02 - 01.03	Fordypelse i AI modeller og design av brukergrensesnitt og system	Wireframes, Azure, fine-tuning, burndown diagram
3	04.03 - 08.03	Valg av AI-modell og brukertesting testing	DSPy, Llama 2, semistrukturert intervju, brukertesting
4	11.03 - 22.03	Gjennomføring av fine-tuning	Hugging Face, SFT Trainer, Google Colab, Llama CPP
5	25.03 - 05.04	Testing av maskinvare og alternativ AI-rammeverk (RAG)	RAG, Pinecone, Langchain
6	08.04 - 19.04	Ferdig prototype	Systemintegrasjon, testplaner, feilretting
7	22.04 - 03.05	Prompt engineering	Forbedring av uthenting gjennom promt engineering
8	06.05 - 17.05	Sluttføring og dokumentasjon, evaluering og brukertesting	Fullføring av dokumentasjon, endelig rapportskriving, evaluering av “chunk” størrelser og brukertesting

Tabell 2: Oversikt over sprinter

3.1 Prosjektstart (Pre-sprint)

Starten av prosjektet var da gruppen fikk bekreftet at samarbeidet med UiA var en realitet på første møte med oppdragsgiver, den 17. januar. Selve prosjektbeskrivelsen var fortsatt under utvikling og ble kun kommunisert muntlig. Likevel måtte gruppen sette rammene for prosjektstyringen og forventningene innad i gruppen.

Gruppen satte klare krav og forventninger til hverandre ved å signere en gruppekontrakt. I tillegg var det viktig at gruppen kom til enighet om hva kvalitet innebærer i prosjektet. Begrepet **kvalitet** kan ha ulike definisjoner basert på semantikk, men Norsk Standard, NS-EN ISO 9000 definerer kvalitet som hvor mye egenskaper eller karakteristikker ved et produkt, tjeneste eller prosess måler seg opp mot behov, krav eller forventninger som er angitt (Gundersen & Halbo, 2018). Ved å sette krav og forventninger til sprinten blir det en felles indikasjon på hvordan kvalitet kan måles og oppnås i gruppen.

3.1.1 Scrum

I et prosjekt er valget av riktig arbeidsmetodikk blitt en nøkkelfaktor for å sikre effektivitet, fleksibilitet og kvalitet i prosjektleveransene. Det er viktig å anerkjenne at ikke alle prosjekter og grupper er like, og det finnes derfor ingen enkeltstående arbeidsmetodikk som passer perfekt for enhver situasjon (Adobe Experience Cloud Team, 2023). I noen tilfeller kan prosjekter og grupper være så unike at det ikke er praktisk å følge en standardisert arbeidsmetodikk som Scrum, Kanban eller Waterfall. Dette kan skyldes en rekke faktorer, inkludert prosjektets størrelse, kompleksitet, bransje, organisasjonsstruktur og kultur, samt tilgjengelige ressurser og kompetanse innenfor gruppen. Uansett hvilken tilnærming som velges, er det viktig å forstå at suksessen til enhver arbeidsmetodikk avhenger av hvordan den tilpasses og implementeres i prosjektet.

Gruppen har valgt å bruke Scrum basert på metodikkens relasjon til *Det Agile Manifestet*. Manifestet ble formulert av 17 systemutviklere, og fremhever viktigheten av individer og samspill over prosesser og verktøy, fungerende programvare over omfattende dokumentasjon, kundesamarbeid over kontraktsforhandlinger og evnen til å respondere på endringer over å følge en plan (Beck et al., 2001).

Scrum er i tillegg valgt på grunn av tidligere erfaring fra IS-200. Arbeidsmetodikken er åpen for agile prinsipper, og legger vekt på åpen kommunikasjon og samarbeid (Schwaber & Sutherland, 2020). Dette understreker viktigheten av individuell interaksjon og samspill, og skjer gjennom Scrum aktivitetene Daily Scrum, Sprint Planning og Retrospective. Videre har man aktiviteten Sprint review, som fremmer samarbeid med kunden ved å demonstrere funksjonalitet og motta tilbakemeldinger, noe som samsvarer med *Det Agile Manifestets* verdi om å samarbeide med kunden gjennom hele prosessen (Beck et al., 2001).

Når det gjelder struktur og roller i gruppen har det blitt bestemt av gruppen at det skal være en flat struktur, og det ble valgt gruppeleder, vara og scrum-master. Gruppelederen og vara fikk ansvar for det administrative arbeidet, deriblant booking av møter, mens Scrum-master fikk ansvar for Scrum-aktivitetene. Her er en oversikt over Scrum-aktivitetene i følgende sprinter:

Daily scrum blir brukt for å holde alle i gruppen oppdatert og ta opp eventuelle utfordringer som noen har støtt på.
Sprint planning er når gruppen setter mål for sprinten og estimerer oppgavene som skal gjennomføres.
Sprint gjennomføring er perioden hvor teamet jobber med oppgavene som ble identifisert under Sprint planningen. Det inkluderer utvikling av funksjonalitet, testing, og andre oppgaver som er nødvendige for å fullføre de planlagte arbeidsoppgavene.
Sprint reviews tillater gruppen å vise oppdragsgiver og veileder arbeidet som blir gjennomført i løpet av sprinten. Dette gir dem muligheten til å gi verdifull tilbakemelding og godkjenning på fremgang og metodikker. Ved å inkludere både oppdragsgiver og veileder i disse møtene blir det også skapt en gjensidig forståelse og tillit mellom gruppen og oppdragsgiveren.
Sprint retrospective er når sprinten blir konkludert fra gruppens perspektiv. Her vil gruppen ta en gjennomgang av hva som gikk bra i sprinten, og hva som kan forbedres til neste sprint (Scrum.org, u.å.).

3.2 Sprint 1

(23.01 - 16.02)

Den første sprinten ble dedikert til undersøkelse og forståelse av systemet.

Verktøy og metoder for denne sprinten:

- Tankemodell
- Systemarkitektur
- Kunnskapsbasen
- Systemdefinisjon
- Use-Case diagram
- Brukerhistorier
- MoSCoW
- Estimering
- Risikovurdering

3.2.1 Sprint planning

Gruppen bestemte seg for å dedikere første sprint til å skaffe mer kontekst og kunnskap innenfor domenene i prosjektet. Målet med sprinten var å få oversikt over systemet og hvem brukerne er. Etter første møte med oppdragsgiver var det usikkerheter og mangel på informasjon av systemet som gruppen ville avklare. Det ble bestemt av gruppen at den første sprinten skulle vare i fire uker. Sprintlengdene ble først definert da vi fikk i oppgave av oppdragsgiver og veileder om å utarbeide en tidsplan for hele semesteret (se vedlegg 5). For at gruppen kunne avklare sine egne forventninger og samtidig oppnå oppdragsgiverens forventninger, var det nødvendig for gruppen å gi et estimat på gruppens fremgang.

3.2.2 Sprint gjennomføring

Før gruppen kunne definere systemet, var det avgjørende å samle relevant data og innsikt i systemets struktur og funksjonaliteter. Møter med oppdragsgiver var gruppens tilnærming til datainnsamling av systemet denne sprinten. Videre presiserte gruppen systemdefinisjonen og analyserte risikoen involvert i prosjektet.

På det første møtet med oppdragsgiver og veileder fikk gruppen beskjed om å konstruere en tankemodell til neste møte (se vedlegg 6), slik at de kunne se hvordan vi tolket domenet. En **tankemodell** er en av flere typer konseptuelle modeller som finnes. “Mental models”, som David Benyon kaller det (Benyon, 2019, s. 31), illustrerer noens forståelse av et konsept, som i gruppens tilfelle er FS-Hjelp og entitetene rundt. Viktigheten av en god tankemodell beskriver Benyon slik: “Hvis folk ikke har en god mental modell av noe, kan de bare utføre handlinger utenat. Hvis noe går galt vil de ikke vite hvorfor og vil ikke kunne komme seg tilbake”.

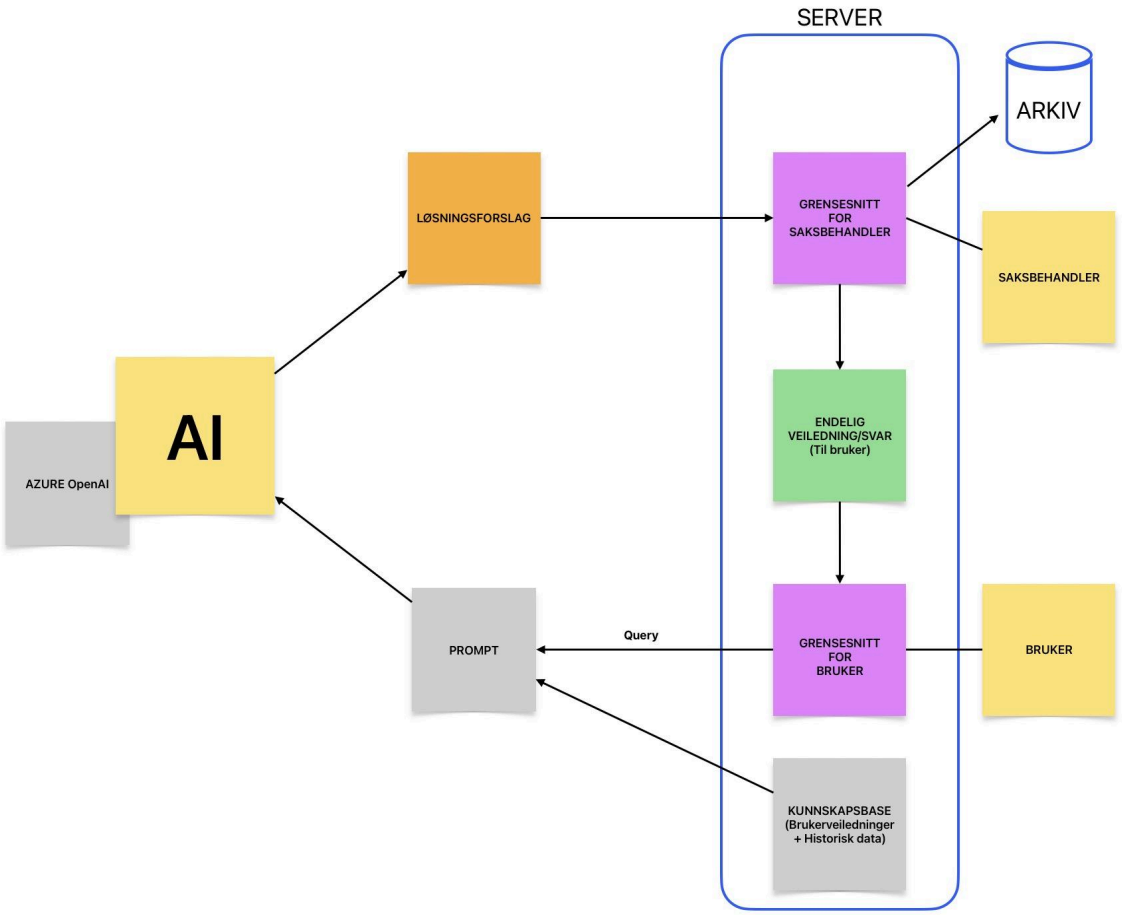
I det første statusmøte ble det presentert førsteutkastet av tankemodellen, og der ble det tydelig at det var muligheter for forbedringer. Derfor utarbeidet gruppen en oppdatert versjon av tankemodellen, som bygget videre på og forbedret den foregående. Den oppdaterte versjonen av tankemodellen fungerer som en systemarkitektur for hele oppbyggingen av det som skulle bli AI-assistenten til FS-Hjelp.

I **systemarkitekturen,** figur 1, ser man oversikten over de forskjellige byggeklossene som skulle drive FS-Hjelp. Komponentene er fargekodet, der de gule lappene er entitetene i systemet, de lilla lappene er brukergrensesnitt, og resten av komponentene er bak kulissene.

Løpet startet med brukeren av FS-Hjelp, som var koblet på systemet via et brukergrensesnitt som lå på en webserver. Der sendte brukeren spørsmål til AI-en via en API nøkkel på webserveren. API står for Application Programming Interface, og er en kode som brukes for å utveksle data mellom to forskjellige systemer eller apper (Aspelund, 2024). Nøkkelen ga oss autentiseringen til å sende AI-en spørringer fra et eksternt system, og det var spesielt

understreket av arbeidsgiver at denne skulle være godt gjemt slik at ondsinnede aktører ikke skal kunne fange den opp.

Spørsmålet fra brukeren ville blitt sendt sammen med informasjon fra vår kunnskapsbase for å danne en effektiv spørring til AI-systemet. **Kunnskapsbasen** er lagret i saksbehandlingssystemet ServiceNow, og inneholder alle brukerveiledningene fra UiAs FS-Hjelp samt historiske data fra tidligere henvendelser, som også var tilgjengelige i ServiceNow sin selvbetjeningsportal. Denne informasjonen var nødvendig for at AI-systemet skulle kunne identifisere riktig veiledning og formulere en løsning for brukeren. Etter å ha blitt generert, ville svaret først bli vurdert av saksbehandleren før det eventuelt ble videresendt til brukeren. Kopier av alle brukerhenvendelser og svar skulle også bli sendt til ServiceNow sitt arkiv periodisk.



Figur 1: Systemarkitektur versjon 1

Den utviklede systemarkitekturen vil bidra til etableringen av **systemdefinisjonen**. En systemdefinisjon defineres av Mathiassen som en konsis beskrivelse av et system uttrykt i naturlig språk (Mathiassen et.al., 2018, s.24). Dette er kritisk for prosjektet i utviklingsfasen

når det er uklart hva som er nødvendig i systemet. I tillegg er det viktig slik at alle i gruppen har en felles forståelse av systemdefinisjonen. For å komme frem til en systemdefinisjon har gruppen valgt å benytte seg av FACTOR-kriteriene som peker på seks viktige faktorer i en systemdefinisjon: funksjonalitet, applikasjonsdomene, forhold (condition), teknologi, objekter og ansvar (responsibility). I tabell 3 har gruppen utfylt FACTOR-kriteriene til systemet.

FACTOR-kriterier

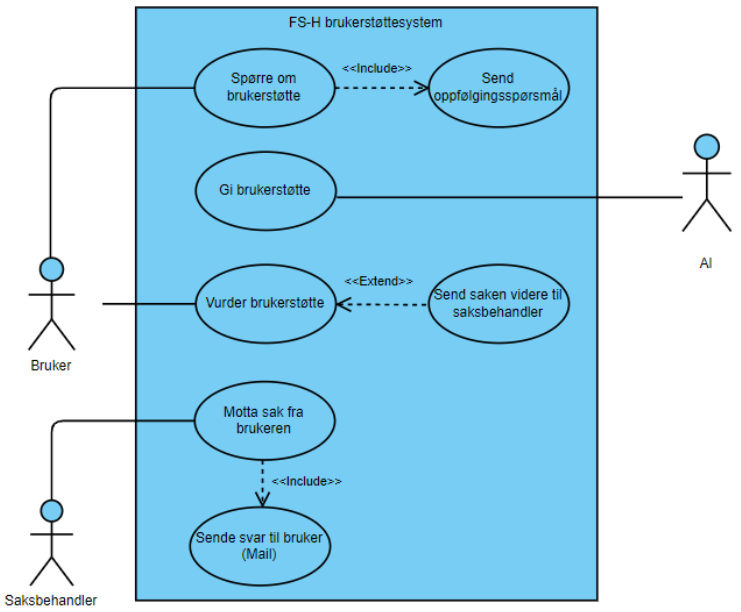
Funksjonalitet	Gi brukerveiledning til brukere av FS.
Applikasjonsdomene	Brukerstøtte, FS-Hjelp.
Forhold	Uregelmessige brukere (occasional users). Offentlig sektor og administrasjonen.
Teknologi	Azure OpenAI GPT 3.5 Turbo, API, python.
Objekter	Kunnskapsbase (brukerveiledninger og historisk data) og stor språkmodell (LLM).
Ansvar	AI-assistent.

Tabell 3: FACTOR-kriterier

Gruppen kom da frem til systemdefinisjonen:

"*En AI-assistent for brukere av FS på UiA. Systemet skal bestå av en LLM som baserer seg på en kunnskapsbase med brukerveiledninger og historisk data til å gi støtte til administrasjonen på UiA. Målet med systemet er å anvende AI-teknologi til å forbedre brukeropplevelsen av brukerstøtte til FS-systemet.*"

For videre analyse har gruppen utarbeidet et **use case diagram**. Et use case diagram illustrerer brukerens mulige interaksjoner med systemet (Mathiassen et.al., 2018, s.122). Diagrammet fremhever ulike use cases og alle typer brukere systemet har. Figur 2 viser at systemet blir brukt av tre typer brukere eller roller: bruker, AI og saksbehandler. Hver rolle i systemet har separate formål og interesseområder. En bruker ønsker å spørre om brukerstøtte til det de sliter med på FS. De har også mulighet til å sende oppfølgingsspørsmål og vurdere brukerstøtten. Dersom det trengs kan de også sende saken videre til en saksbehandler fra FS-Hjelp. AI-en vil bare generere brukerstøtte og svar til det brukeren spør om, mens saksbehandler tar imot saker som ikke kan bli løst av AI.



Figur 2: Use case diagram

Som et resultat av å definere systemet og avklare systemets roller og deres use cases, kan gruppen gå videre til å beskrive oppgaver gjennom brukerhistorier. **Brukerhistoriene** er enkle beskrivelser av ønsket funksjonalitet som gruppen har estimert med historiepoeng (story points) basert på enighet (Adams, 2021; Mikalsen, 2024). Backlog items i prosjektet vil bli basert på brukerhistoriene. Gruppen har også bestemt å prioritere oppgavene ved bruk av **MoSCoW-metoden**. MoSCoW-metoden står for "Must have", "Should have", "Could have" og "Won't have". Disse kategoriene blir da brukt for prioritering av backloggen. Tabell 4 viser et utsnitt av MoSCoW-prioriteringen av brukerhistoriene. Hele tabellen kan finnes i vedlegg 7.

Som en …	**Ønsker jeg å…**	**Slik at …**	MoSCoW (prioritering)
Bruker	Spørre om brukerstøtte	Jeg kan å løse utfordringene jeg har med FS	Must have
AI	Gi brukerstøtte	Jeg kan løse utfordringene til brukeren uten ventetid på saksbehandler	Must have
Bruker	Sende oppfølgingsspørsmål	Jeg kan sende saken videre til saksbehandler dersom brukerstøtten fra AI ikke hjelper	Should have
Saksbehandler	Motta sak fra bruker	Jeg kan gi de brukerstøtten	Could have

		som AI ikke kan løse	

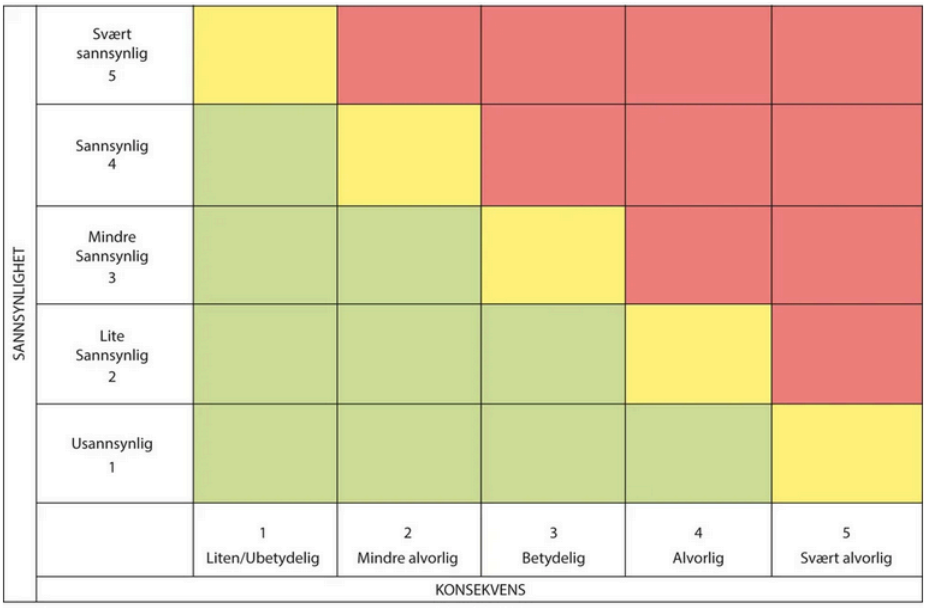
Tabell 4: Utsnitt av brukerhistorier og MoSCoW-prioritering.

Å estimere tidsbruk i prosjekter er krevende, men nødvendig for å planlegge og allokere ressurser slik at man kan beregne eller sette mål til når man ønsker å være ferdig. Man må også være klar over at estimeringer bare er antakelser og ikke noe som er deterministisk. Historiepoeng er abstrakte enheter basert på konsensus, som representerer kompleksitet, usikkerhet eller størrelse i oppgaven som skal gjennomføres (Radigan, u.å.). Formålet med å måle arbeid i historiepoeng er ikke for å estimere produktivitet, men heller innsatsen som kreves.

Backloggen ble ført opp i styringsverktøyet "ClickUp". "ClickUp" er en tjenesteplattform for produktivitet og prosjektstyring. Styringsverktøyet ble brukt for å gjennomføre og få oversikt over Scrum-elementene i prosjektet. Verktøyet ble brukt til å planlegge Product - og Sprint backlog. Det ble også mulig å prioritere oppgaver og måle tidsbruk på oppgavene.

Etter å ha definert systemet er det nødvendig å gå videre til identifisering av potensielle risikoer som kan påvirke prosjektets fremgang. Å forstå systemet ga gruppen mulighet til å identifisere potensielle utfordringer og usikkerheter som kunne oppstå underveis. Risikoanalysen ble derfor et naturlig neste skritt for å kartlegge og evaluere disse risikoene.

Risiko er et begrep som omhandler fremtidige konsekvenser av en handling eller usikkerhet, der konsekvensene kan være avvik i målsetting (Aven, 2018). Derfor ønsket gruppen å utføre en risikoanalyse, se vedlegg 8, for å vurdere sannsynligheten og konsekvensene for mulige risikoer i prosjektet. Ved å identifisere disse risikoene ville noen av usikkerhetene i prosjektet oppheves, og risikoen for at prosjektet feiles reduseres. Gruppen har gjennomført en risikoanalyse ved bruk av risikomatrisen for å illustrere sannsynligheten og konsekvensene av risikoene. Et utsnitt av dette kan observeres i tabell 5. Risikomatrisen, figur 3, er et diagram som består av to akser hvor X-aksen tar for seg konsekvens for gitte hendelser, mens Y-aksen inneholder tilhørende sannsynlighet (Aven, 2022). Konsekvens og sannsynlighet måles i denne matrisen fra 1 til 5.



Figur 3: Risikomatrise (Regjeringen, 2016)

Risiko	Konsekvens (1 til 5)	Sannsynlig-het (1 til 5)	Risiko	Tiltak for at ikke skjer	Tiltak for å redusere skade
API-nøkkel blir fanget opp	4	2	8	Skjule API-nøkkelen og rotere nøkkel periodisk	Slette API-nøkkelen og erstatte den med en ny
AI-assistent gir dårlig støtte	3	2	6	Gi LLM kvalitetsdata og gjennomfør tilstrekkelig testing	Mulighet til å følge opp på spørsmål, eller gå videre til en saksbehandler
Manglende kunnskap for å fullføre prosjektet	4	5	20	Kontinuerlig læring innenfor domenet., Spørre om hjelp fra veileder. Hjelpe hverandre i gruppen	Tilby tilstrekkelig med dokumentasjon på hva som er gjort så prosjektet kan videreutvikles
Manglende dialog med oppdragsgiver	4	1	4	Holde god kontakt med oppdragsgiver og fortsette med ukentlige møter	Sette opp møter og gjenopprette dialog

Tabell 5: Utsnitt av risikomatrisen (Vedlegg 8)

3.2.3 Sprint review

Første Sprint review ble gjennomført med deltakelse fra gruppemedlemmer, oppdragsgiver og veileder. Fokuset på møtet var å vise fram produkt-backloggen som inneholdt prioriteringer og estimeringer av brukerhistoriene. Oppdragsgiver var enig i prioriteringene og estimeringene. I tillegg var oppdragsgiver og veileder fornøyd med fremgangen og starten på prosjektet, hvor gruppen hadde fått avklart forståelse av domenet og lagt et grunnlag for kommende sprinter.

Oppdragsgiver identifiserte forbedringspotensial i dokumentasjon av gjennomføringen. Det var dokumentert hva som var gjort, men ikke loggført tidsbruk og -fordeling. I tillegg var det en del usikkerhet knyttet til prosjektet og oppgavene, noe som indikerte at sprintens lengde var for lang.

3.2.4 Sprint retrospective

For å forbedre dokumentasjonen av vårt arbeid besluttet gruppen, etter anbefaling fra veilederen, å implementere individuell loggføring i dagbok i den kommende sprinten. Dette tiltaket ville sannsynligvis bidra til å sikre kvaliteten i prosjektet. Ved å ha en dagbok tilgjengelig fra alle på gruppen, ble det mulig å se tilbake og forstå tankene og beslutningene som ble tatt på det aktuelle tidspunktet i prosjektet.

For å sikre kvalitet og fremgang i prosjektet holdt vi jevnlige samtaler med oppdragsgiver for å tydeliggjøre forventninger og viktighetsgrad av komponenter. I tillegg gjennomførte vi regelmessige gjennomganger av våre utforskninger, og dokumenterte vår læring gjennom konseptuelle modeller. Disse tiltakene tillot oss å demonstrere en tydelig fremgang og forståelse, selv i den tidlige fasen av prosjektet, og sikre oppdragsgiverens tillit.

Gitt at oppgavene i sprinten var uklare for gruppen på forhånd, burde sprintlengden vært kortere. Både emneansvarlig og veileder understreket at desto mer uklarheter det er i sprintene, desto kortere bør de være. Dette skyldtes behovet for å unngå mye tidsbruk på det uklare, samtidig som man kunne reflektere og justere kursen dersom ting ikke gikk som planlagt.

Totalt sett kom gruppen så langt man kunne med ressursene som var tilgjengelig, og målene for sprinten ble oppnådd.

3.3 Sprint 2

(19.02-01.03)
Den andre sprinten ble dedikert til fordypelse i AI modeller og design av brukergrensesnitt og system mens gruppen ventet på ressurser fra IT-avdelingen.
Verktøy og rammeverk for denne sprinten:

- Jira
- Wireframes
- Microsoft Azure
- Fine-tuning
- Jupyter Notebook
- Burndown diagram
- Systemarkitektur versjon 2

3.3.1 Sprint planning

Etter veileders råd og etter å ha reflektert over *"Statusmøte 1*", hvor alle bachelorgruppene presenterte sine prosjekter, samt gjennomført Sprint Retrospective 1, skulle vi i denne sprinten starte med å føre individuelt arbeid i dagbok. "ClickUp", som var verktøyet vårt for backlog i den første sprinten, påløpte seg plutselige økonomiske kostnader og dermed skulle styringsverktøyet endres til Jira i neste sprint.

Under sprinten måtte vi også planlegge å utnytte tiden mens vi ventet på svar fra IT-avdelingen angående API-en til AI modellen fra UiA sitt Azure Workspace, samt kunnskapsartiklene fra ServiceNow, som inneholdt veiledninger og historisk data. Med ventetiden, som kunne strekke seg fra noen få dager til flere uker, besluttet vi å fokusere på utviklingen av wireframes og frontend, samtidig som vi utarbeidet en intervjuguide for et semistrukturert intervju og brukertesting med en erfaren bruker av FS, dersom tiden tillot det. Dette var viktig for å få innblikk og perspektiver fra noen med praktisk erfaring i FS, noe vi selv ikke hadde tilgang til som studenter.

3.3.2 Sprint gjennomføring

Design av brukergrensesnittet var en stor del av Sprint 2 som kom gjennom prosessen med **wireframes.** Wireframes utforsker designeres konseptuelle skisser av interaktive produkter, og "er skisser av strukturen til et programvaresystem" (Benyon, 2019, s. 194). Wireframes fungerer som en designplan eller et skjelett for en mulig løsning, og de kan utvikles i enten low- eller high-fidelity versjoner. Mens prototyper fokuserer på interaktivitet og etterligner det endelige produktet nøyaktig, har wireframes begrenset interaktivitet og legger hovedvekt på strukturen og plasseringen av elementer i grensesnittet. I vårt prosjekt, hvor målet var å utvikle en AI-assistent, brukte vi Figma for å lage high-fidelity wireframes. Dette ga både gruppen, oppdragsgiver og det fremtidige intervjuobjektet et klart bilde av hvordan AI-assistenten ville se ut (se figur 4 og figur 5 under).

Ettersom prototypen ble basert på funksjonene fra brukerhistoriene som ble utarbeidet i forrige sprint, ga den oss verdifulle innsikter før det faktiske produktarbeidet startet. I forberedelser av det semistrukturerte intervjuet som skulle utføres i Sprint 3 utarbeidet gruppen en intervjuguide (vedlegg 10) i denne sprinten.

Figur 4: Saksside av Figma wireframe (Versjon 1)

Figur 5: Chatside av Figma wireframe (Versjon 1)

Midt i sprinten fikk vi tilgang til API nøkkelen fra IT-avdelingen på UiA, som vi hadde ventet på for å bevege oss fremover på det tekniske aspektet. Det var da vi fikk testet fine-tuning i **Microsoft Azure** sitt Deployment Environment. **Fine-tuning** refererer til prosessen der en allerede opplært AI-modell, som i vårt prosjekt var GPT-3.5 Turbo, justeres videre med et mindre datasett for å tilpasse modellens ytelse til en spesifikk oppgave eller et spesifikt domene. Dette innebærer å justere modellens vekter og parametere slik at den bedre kan forutsi eller analysere data som er nært relatert til de spesifikke kravene til bruksområdet

(IBM, u.å.c). Denne teknikken tillater derfor å tilpasse en språkmodell til eget bruk uten de enorme kostnadene av å trene en stor språkmodell fra bunnen av.

For å kjøre en initiell test på vår fine-tuned modell lagde gruppen et treningsdatasett og et valideringsdatasett på ti eksempler hver, basert på de eksisterende veiledningene til FS på ServiceNow. De to datasettene ble satt opp i JSONL format og deretter lastet inn, prosessert og opplastet til Azure plattformen gjennom Jupyter Notebook. Videre, ved bruk av en API-nøkkel og et Azure endepunkt lagret som miljøvariabler, var neste steg å bruke datasettene til å fine-tune modellen for å teste den med spørsmål knyttet til veiledningen av FS-systemet. For å gjøre dette måtte IT-avdelingen på UiA distribuere vår fine-tuned modell, ettersom gruppen ikke fikk tillatelse til Azure plattformen, og det ble da kjørt en suksessfull initiell test av den fine-tunede AI-modellen.

Videre fant gruppen ut fra Azure sine nettsider og dokumentasjon at treningen gjennom denne modellen påløpte seg kostnader for hver time som gikk og for hver 1000 tokens som ble brukt. Spørsmålet gruppen stilte seg da var om dette ville være økonomisk bærekraftig for UiA å implementere. Derfor ble dette tatt videre til oppdragsgiver. En oversikt over Microsofts priser ligger under i figur 6.

Fine-tuning models

Models	Training per compute hour	Hosting per hour	Input Usage per 1,000 tokens	Output Usage per 1,000 tokens
Babbage-002	N/A	N/A	N/A	N/A
Davinci-002	N/A	N/A	N/A	N/A
GPT-3.5-Turbo (4K)	$45	$3	$0.0015	$0.002
GPT-3.5-Turbo (16K)	$68	$3	$0.003	$0.004

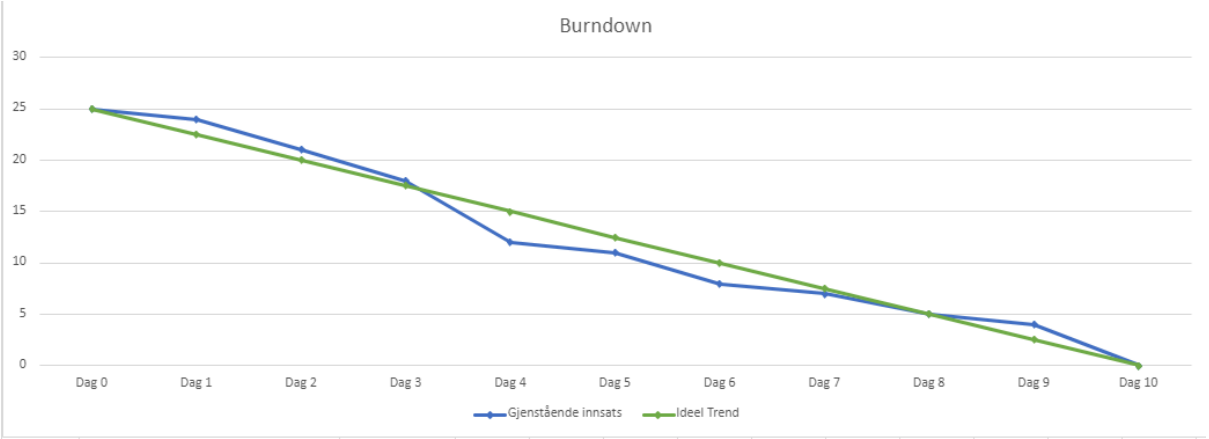
Figur 6: Pris over Azure Fine-tuning modeller (Azure OpenAI Service - Pricing | Microsoft Azure, u.å.)

Noe gruppen gjorde for å være åpne for andre løsninger for valg av AI modeller og biblioteker var å kommunisere med en annen bachelorgruppe, Gruppe 24 (SmartOrg). De hadde også blitt anbefalt av emneansvarlig å snakke med oss om dette temaet, ettersom vi hadde ganske like mål og rammer rundt prosjektene våre. Denne kommunikasjonen resulterte dessverre i ingen læringsutbytte, men mer en forståelse av SmartOrgs situasjon.

I det tredje ukentlige statusmøtet ble vi bedt om å sette opp et **burndown diagram**. Dette er et visuelt verktøy som brukes i prosjektstyring, spesielt i agile metodikker som i vårt tilfelle med Scrum rammeverket (ServiceNow, u.å.).
Det er flere fordeler med å lage et burndown chart, og dette er tre av de mest relevante årsakene i vårt tilfelle:

- Det er et nyttig verktøy for å kommunisere fremdrift og status til aktørene utenfor teamet, i vårt tilfelle oppdragsgiver og veileder.
- Det gir et visuelt bilde av hvordan arbeidet går gjennom sprintene. Teamet kan se om vi er på rett spor for å fullføre oppgavene innenfor tidsrammen.
- Eventuelle avvik mellom den planlagte fremdriften og den faktiske fremdriften kan oppdages tidlig. Dette gjør det mulig å identifisere problemer eller hindringer raskt og ta nødvendige tiltak for å løse dem.

Figur 7 illustrerer burndown diagrammet for Sprint 2 , hvor Y-aksen er historiepoengene, mens X-aksen er tidslinjen for sprinten.



Figur 7: Burndown chart for Sprint 2

3.3.3 Sprint review

Andre Sprint review ble gjennomført med oppdragsgiver og veileder. I møtet ble det vist hva som ble gjennomført i sprinten. Gruppen viste først frem wireframes som ble utviklet, og at det har blitt klargjort et semistrukturert intervju, med brukertesting (vedlegg 10). Oppdragsgiver videresendte kontaktinformasjon til et potensielt intervju- og testobjekt.

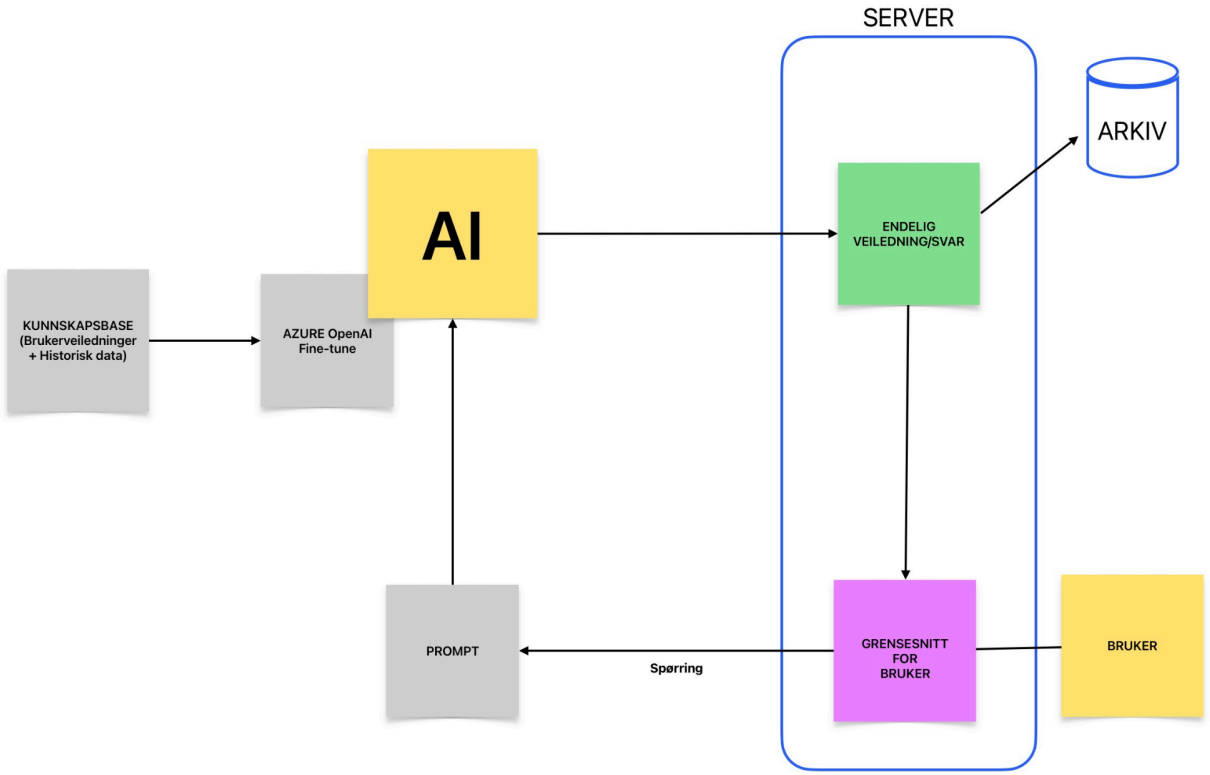
Gruppen diskuterte også usikkerhet rundt kostnader med oppdragsgiver, og fikk beskjed om å sende e-post til IT-avdelingen for å avklare dette. Dette var også da oppdragsgiver ønsket at gruppen skulle fokusere mer på det tekniske, og sette brukergrensesnittet på vent.

3.3.4 Sprint retrospective

Elementene i Sprint backloggen for denne sprinten var mer konkrete og realistiske, noe som også var et av målene for forbedring til gruppen fra Sprint 1 retrospective.
Gruppen er også fornøyde med dokumentasjonen i denne sprinten, hvor vi innførte dagbok, noe som skal tas med i videre sprinter. Et utsnitt av sprintens dagbok kan finnes i vedlegg 9.

Fra forrige tankemodell i Sprint 1 (se figur 1), har det blitt noen endringer i systemarkitekturen. Som vist i figur 8 nedenfor har kunnskapsbasen blitt flyttet ut av

FS-systemet. Dette er fordi gruppen gikk for en fine-tunet modell. Som nevnt tidligere i sprinten betyr det at AI-modellen allerede er trent opp på kunnskapsbasen, og informasjonen er dermed ikke nødvendig i spørringene fra brukeren. Grensesnittet for saksbehandler ble også fjernet. Etter diskusjon med oppdragsgiver ble det konkludert at saksbehandler skulle bruke samme grensesnitt som en vanlig bruker. Hvis svarene ikke var opp til standard og trengte vurdering før videresending, ville saksbehandlerne være de eneste brukerne av systemet, og bruke det som hjelpemiddel for å besvare henvendelser via e-post, som før.



Figur 8: Systemarkitektur versjon 2

3.4 Sprint 3

(04.03 - 08.03)

I den tredje sprinten hadde gruppen ett hovedmål: å velge veien videre med tanke på AI.

Verktøy og rammeverk for denne sprinten:

- DSPy
- Llama 2
- Semistrukturert intervju
- Brukertesting

- Systemarkitektur versjon 3

3.4.1 Sprint planning

Sprintlengden for Sprint 3 ble satt av gruppen på kun én uke. Dette var fordi gruppen kun hadde ett mål med sprinten, å velge veien videre med tanke på AI, og dette var forventet å bli fullført innen denne tiden.

Oppdragsgiver ønsket i forrige Sprint review at gruppen skulle rette seg mer mot det tekniske, og ble tipset av veileder om å se på DSPy og Llama 2 med tanke på at gruppen måtte gå over til en "Open Source" språkmodell. I denne perioden var DSPy fremdeles nytt. Dette betydde at verken veileder eller gruppen hadde kunnskap om hva DSPy gjorde og heller ikke hva det ble brukt til. Til tross for dette virket det på overflaten som at det hadde potensial for gruppens oppgave, og derfor valgte gruppen å rette fokuset mer mot dette i denne sprinten.

3.4.2 Sprint gjennomføring

I begynnelsen av denne sprinten fikk gruppen tilbakemelding fra oppdragsgiver på at det ikke var økonomisk bærekraftig å fortsette å bruke Microsoft Azure, noe som tidligere ble tatt opp med oppdragsgiver i Sprint 2. Dette gjorde at gruppen måtte gå over til å kun fokusere på modeller som kunne kjøres på lokale maskiner (Open Source løsninger). Det betyr at den valgte språkmodellen måtte være kompatibel med maskinvarene gruppen hadde tilgjengelig, men fremdeles kunne svare tilfredsstillende på sakene. Tilfredsstillende vil si at svarene er basert på FS sin kunnskapsbase.

DSPy er et rammeverk for å optimalisere prompts og svar fra språkmodeller. Her brukes maskinlæring til å skape en mer systematisk måte å sette sammen flere prompts. Dermed blir det enklere å instruere språkmodellen etter egendefinerte spesifikasjoner. Dette gjøres uten å måtte oppdatere beregningene til språkmodellen, noe som kan medføre betydelig tidsbruk og kostnader. DSPy tilbyr derfor å oppnå de samme resultatene, men utføres ved bruk av algoritmiske instruksjoner. Disse instruksjonene innebærer at programmet som brukes til å drive språkmodellen holdes separat fra språkmodellen. Dette medfører at språkmodellen ikke påvirkes av programmet, og instruksjonene skjer derfor bare på et overfladisk nivå. DSPy kan brukes til kraftige modeller, som GPT-3.5 og GPT-4, men også lokale modeller som Llama 2 (DSPy, u.å.).

DSPy tilbyr mange komplekse funksjoner som ikke var nødvendig for gruppens oppgave, og derfor bestemte gruppen å ikke bruke mer tid på å forstå alle funksjonene til DSPy. Gruppen valgte da å fortsette med fine-tuning på lokal språkmodell. For å oppnå dette måtte gruppen se på Llama 2.

Språkmodellen **Llama 2** er utviklet og utgitt gjennom et samarbeid med Meta og Microsoft. Utvikling av språkmodeller krever mye data, noe som dette samarbeidet tar nytte av ved å ta i bruk data fra akademiske og industrielle institusjoner. På grunnlag av at dataene blir hentet fra brede kilder, valgte Meta å åpne denne språkmodellen for akademisk og kommersiell bruk. Derfor er Llama 2 tilgjengelig på delingsplattformen Hugging Face og gjennom skybaserte applikasjoner via Microsoft Azure. For å kunne drive lokale språkmodeller kreves det gunstig maskinvare. Meta har derfor lansert tre ulike versjoner av Llama 2 der størrelsen på modellen varierer fra 7B opp til 70B. Dette tilsier hvor mange parametre modellen innebærer i milliarder. Det er disse parametrene som gir språkmodeller evnen til å lære komplekse mønstre og relasjoner fra tekstdata. Llama 2 er bare trent på engelske data, og kan derfor bare svare på engelsk (Meta, 2023). For at den skulle kunne svare på norsk, måtte gruppen ta i bruk en allerede fintunet Llama 2 modell som har blitt trent på norske data. Denne modellen er trent av Ruter AS.

Parallelt med AI valget som skulle gjøres, gjennomførte gruppen et **semistrukturert intervju**. Dette var gruppens første tilnærming til datainnsamling. Semistrukturert intervju er en kvalitativ metode for å samle data, hvor man intervjuer én person om gangen for å utforske et spesifikt problem grundig. Hensikten med metoden er å få innsikt i intervjuobjektets synspunkter angående en spesifikk hendelse eller situasjon, som i vårt tilfelle var AI-assistenten knyttet til FS-hjelp (Rutledge & Hogg, 2020, s. 1; Sander, 2023).

Det første intervjuet gruppen gjennomførte ble kombinert med **brukertesting**, da intervjuobjektet trengte å se wireframene våre for å kunne svare på flere av spørsmålene. Brukertesting er en metode som brukes for å demonstrere bruken av et produkt for brukeren, slik at man kan observere hvordan de reagerer og samhandler med det. På denne måten kunne vi ikke bare innhente intervjuobjektets synspunkter, men også validere våre designvalg og identifisere eventuelle problemer eller forbedringsmuligheter tidlig i prosessen (Usability Testing: Evaluative UX Research Methods, u.å.).

Etter gruppen gjennomførte brukertesten kom det spesifikke endringsforslag til brukergrensesnittet. Disse endringene ble implementert direkte inn i gruppens wireframe på Figma, men på grunnlag av at gruppen måtte være mer tekniske denne sprinten ble ikke dette videreutviklet.

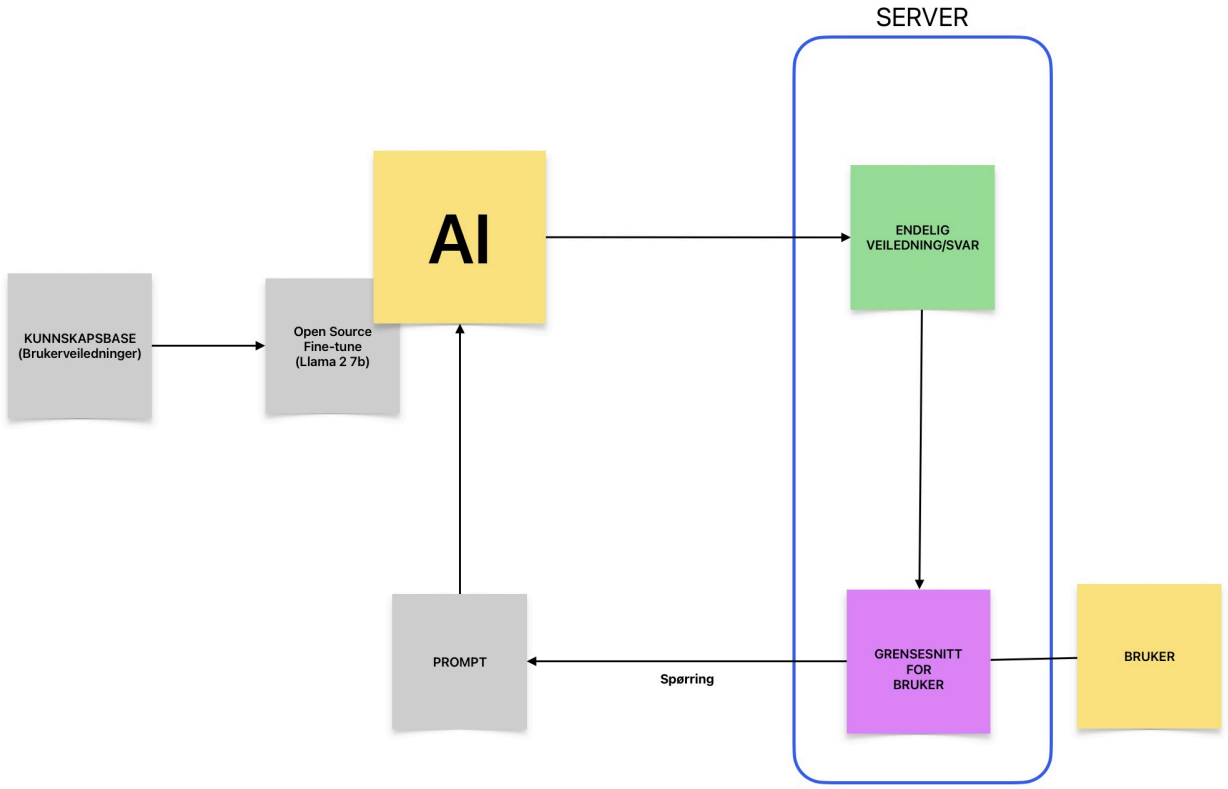
3.4.3 Sprint review

Tredje Sprint review ble gjennomført med oppdragsgiver. Her tok vi opp resultatene av intervjuet, samt “Open Source” løsninger av AI vi hadde sett på. Produktmålene ble også endret i enighet med oppdragsgiver.

3.4.4 Sprint retrospective

Til tross for at gruppen valgte å ikke fortsette med DSPy, fikk gruppen fremdeles ny kunnskap om ulike potensiale løsningsforslag til hvordan språkmodellen kunne svare basert på kunnskapsbasen. Dette gjorde at gruppen fikk bedre forståelse av hvordan språkmodeller og AI generelt fungerer, men DSPy sitt kompleksitet forvirret gruppen angående hvilken teknologi som var nødvendig for oppgaven. Derfor ble det bestemt at gruppen skulle fortsette med fine-tuning, hvor det allerede var bygget kunnskap.

Nedenfor er oversikt over den nåværende systemarkitekturen (se figur 9). Det er tre endringer av systemarkitekturen fra Sprint 2 (se figur 8). Første og viktigste endring er endring av AI-modellen, der Azure OpenAI modellen har blitt erstattet med en Open Source LLM, Llama 2. Den andre endringen, som er sentral for fine-tuning av AI, er fjerningen av historiske data. Dette er grunnet informasjonssikkerhetslederens (CISO) og personvernombudets (PVO) beslutning om at henvendelsene hadde for mye personlig data. Det angikk både gruppemedlemmene og for bruk av fine-tuning. Den siste endringen var fjerning av arkivering til ServiceNow. Arkivering av henvendelser ble fjernet fra systemarkitekturen ettersom scopet av produktet måtte bli innsnevret etter hindring med AI-modell, og arkivering-funksjonen var nedprioritert for gruppens leveranse.



Figur 9: Systemarkitektur versjon 3

3.5 Sprint 4

(11.03 - 22.03)

I den fjerde sprinten har gruppen mål om å gjennomføre fine-tuning.

Verktøy og rammeverk for denne sprinten:

- Hugging Face
- Fine-tuning gjennom SFT Trainer
- Google Colab
- Llama CPP

3.5.1 Sprint planning

I Sprint 4 siktet gruppen på å få tak i bedre maskinvare for å kunne fine-tune. Planen var å høre med både UiA Grimstad og UiA Kristiansand om å låne en datamaskin med tilstrekkelig maskinvare for å kunne teste fine-tuning lokalt. Før gruppen eventuelt fikk tilgang på dette skulle vi teste på egen maskinvare, hvor gruppen måtte forholde seg til skybaserte maskinvaretjenester som Google Colab.

I begynnelsen av sprinten fikk gruppen tilgang til kunnskapsbasen som en PDF-fil, men denne var formulert og satt opp med mange blanke sider. Dette gjorde at kunnskapsbasen ble større enn det den måtte være. For å optimalisere kunnskapsbasen slik at den tok mindre lagringsplass, måtte gruppen manuelt vaske og formatere kunnskapsbasen over til et regneark.

Forventningene gruppen hadde for denne sprinten var å fullføre fine-tuning på en norsk språkmodell, for å så kunne teste den på kunnskapsbasen. For å oppnå dette målet, må gruppen forstå ulike måter å fine-tune språkmodeller slik at vi kan finne den mest relevante metoden for oppgaven.

3.5.2 Sprint gjennomføring

Hugging Face blir hovedsakelig brukt i gruppens oppgave til å laste ned språkmodeller. Den kan sammenliknes med andre skybaserte vertstjenester for programvarer som GitHub. Språkmodellene på Hugging Face er allerede opptrent, og er åpne og tilgjengelig for alle å laste ned. I tillegg er det mulig å laste opp både egne språkmodeller og datasett (Ferrer, 2023). Her har gruppen lastet opp den ferdig formaterte kunnskapsbasen slik at den enkelt kan lastes ned og brukes av fine-tuningsprosessen.

I tillegg til å lagre ulike språkmodeller, tilbyr Hugging Face flere metoder og funksjoner som kan brukes til å fine-tune ferdigtrente språkmodeller. **SFT Trainer** er en enkel metode for å

fine-tune en språkmodell på spesifikke datasett, og står for «Supervised Fine-tuning Trainer». Den gjør det enkelt ved at metoden støtter flere typer datasett. Ulike datasett har som oftest ulike oppsett og stil, og ved bruk av SFT Trainer vil disse ulikhetene være ubetydelige. Fine-tuning på SFT Trainer tilpasser seg ut fra hvor stor datasettet og språkmodellen er. Dette betyr at tiden og maskinvaren som krever for fine-tuning avhenger av disse faktorene (Walker, u.å.). Med tanke på at kunnskapsbasen gruppen har fått ikke er altfor stor sammenliknet med andre kunnskapsbaser på Hugging Face, hadde gruppen forhåpninger om at gjennomføringen av fine-tuningsprosessen var mulig med maskinvaren gruppen har tilgjengelig.

For å overkomme maskinvarebegrensningene, brukte gruppen **Google Colab**. Dette er en skybasert maskinvaretjeneste som tilbyr grafikkprosessorer med CUDA Cores og dedikert minne for grafikkprosessorer, noe som kreves for å fine-tune. CUDA er en forkortelse for Compute Unified Device Architecture, og er NVIDIA sin eksklusive grafikkprosessorskjerner. CUDA Cores er spesifikt laget for å prosessere blant annet AI-relaterte prosesser (Ryles, 2022). Disse er viktig både når gruppen skal fine-tune og kjøre språkmodellen, i og med at fine-tuning er en krevende prosess. Google Colab tilbyr grafikkprosessoren NVIDIA T4 i deres gratisversjon, noe som er tilstrekkelig til gruppens behov. I tillegg tar Google Colab i bruk standard notebook-oppsett som gjør det enkelt å dele og kjøre kode i programmeringsspråket Python (AlmaBetter, u.å.).

Begrensningene i Google Colab gjelder hovedsakelig når det kommer til tidsbruk. Google har satt en begrensing på maksimum 12 timer per kjøring for en notebook (Google. u.å), men ut fra det gruppen har testet, varierer denne begrensingen fra hvor lenge kjøringen gikk. Dersom gruppen brukte Google Colab i flere arbeidsdager kontinuerlig, blir kjøringen gradvis mer begrenset. Fra erfaring, kan kjøringen bli begrenset til kun én time per dag. For at begrensingen skulle oppheves, måtte kjøringen være inaktiv i flere dager.

Begrensningene til Google Colab var for krevende og restriktive til at gruppen kunne fine-tune på denne tjenesten uten avbrytelser. Dermed sendte gruppen en forespørsel til veileder og UiA om tilgjengelig maskinvare, og fikk tildelt en syv år gammel datamaskin. Til tross for alderen var spesifikasjonene til denne maskinen på grensen til å være tilstrekkelig. Det var bekymringer angående at generativ AI ikke var utbredt da denne datamaskinen ble bygget, men ut fra at SFT Trainer kan tilpasse seg til datasettets størrelse og språkmodell, var det potensiale for at fine-tuningsprosessen kunne fullføres. På grunnlag av dette, valgte gruppen å prøve å fine-tune på denne datamaskinen.

Det blir stadig tilgjengeliggjort nye språkmodeller på Hugging Face med ulike styrker og svakheter. For å finne den mest relevante til gruppens fine-tuningsprosess, måtte gruppen gjennomføre tester med flere ulike språkmodeller. Dette kan gjøres på en lokal datamaskin på **Llama CPP,** som er et grensesnitt for å kjøre ferdigtrente språkmodeller fra

kommandolinjen. Den er kompatibel med flere store språkmodeller som Llama 2 og GPT-modeller (AI Verse, 2023). Gruppen fikk til å teste språkmodellen Yi-6B som har blitt trent av 01-AI. For å kunne kjøre denne testen måtte gruppen kompilere kildekoden på en MacBook Pro med prosessoren M1 Pro, som er minimumskrav satt av Yi-6B. Denne testen med Yi-6B var hovedsakelig for å teste om Llama CPP fungerte, slik at gruppen kunne kjøre den endelige fine-tunet språkmodellen med kunnskapsbasen på Llama CPP når denne modellen ble klar. Språkmodellen Yi-6B har ikke evnen til å svare på norsk, og ble dermed ikke relevant for videreutvikling.

3.5.3 Sprint review

Fjerde Sprint review ble gjennomført med oppdragsgiver og veileder. I møtet ble det tatt opp vanskene gruppen hadde med fine-tuning grunnet maskinvare. Dette på både egne stasjoner og maskin lånt fra IT-avdelingen. Fine-tuningen var ikke vellykket i noen av testene, som endte med at målene gruppen hadde satt til seg selv ikke ble nådd. Dette var ikke uforventet for oppdragsgiver, ettersom han hadde blitt informert om maskinvareproblemene tidligere. En annen alternativ løsning ble også tatt opp av gruppen, og hva planen var fremover.

3.5.4 Sprint retrospective

Selv om vi ikke hadde tilgang på tilstrekkelig maskinvare brukte gruppen mye tid på å utforske med fine-tuning. Problemene gruppen møtte på i løpet av fine-tuningsprosessen, som blåskjerm og avbrutte Google Colab sesjoner, tilbakesatt gruppen og vi mistet verdifull tid. Dette var med tanke på at hver gang det oppstod problemer, måtte gruppen som regel starte fine-tuningsprosessen fra begynnelsen, noe som tok opp mot tre timer. Til tross for alle problemene, har det vært tilfeller der fine-tuningsprosessen kom til de siste stegene før det oppstod problemer. Dette førte til at gruppen undersøkte og endret på ulike parametre i håp om at disse kunne rette opp på problemene.

Testing av språkmodeller på Llama CPP gjorde at gruppen bedret kunnskapen om hvordan språkmodeller kjøres på lokale datamaskiner. I tillegg åpnet Llama CPP opp muligheter til å teste ulike språkmodeller. Dette hjalp gruppen med å finne svakheter og styrker ved de forskjellige språkmodellene.

3.6 Sprint 5

(25.03 - 05.04)

Da Sprint 4 var ferdig var det ikke gjort nok testing, og testingen på UiA sin datamaskin måtte fortsette i Sprint 5. Her ble det samtidig gjort analyse og forskning på andre AI rammeverk og

teknikker ettersom risikoanalysen viste at faren for begrensing av maskinvare, knyttet til fine-tuning, var høy.

Verktøy og rammeverk for denne sprinten:

- Retrieval-Augmented Generation (RAG)
- Systemarkitektur versjon 4

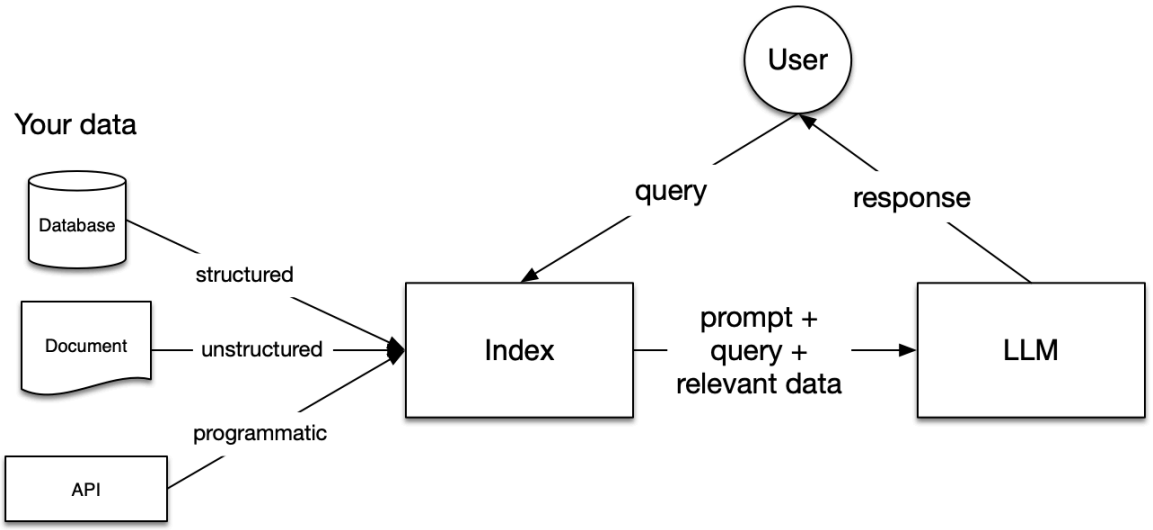
3.6.1 Sprint planning

Denne sprinten startet med et parallelt fokus på testing på UiA sin datamaskin og utviklingen av AI med RAG. Målet med denne sprinten var å få til en vellykket fine-tuning av en AI modell, samtidig ble det undersøkt et komplett skift i systemarkitekturen gjennom implementeringen av RAG rammeverket. Dette skiftet var for å svare på om det fortsatt var mulig å lage en lokal open-source AI-assistent for FS, dersom den begrensede mengden med ressurser som gruppen hadde tilgang på førte til en stopp i fremgangen på fine-tuning.

3.6.2 Sprint gjennomføring

Det todelte fokuset i denne sprinten stammet fra at gruppen lærte i tidligere sprinter om maskinvare nødvendighetene for å kunne kjøre og fine-tune AI modeller lokalt. Dermed visste gruppen i starten av denne sprinten, under testinga på UiA sin datamaskin, at det var stor fare for at det ikke var tilstrekkelig med maskinvareytelse tilgjengelig på maskinen. Derfor ble det satt i gang forskning på RAG rammeverket som et sikkerhetsnett hvis fine-tuningen ikke skulle la seg gjennomføre.

RAG, som er kort for “Retrieval-Augmented Generation”**,** er et rammeverk som tillater AI modellen å hente informasjon fra eksterne kilder, noe som gir den tilgang til domenespesifikk informasjon og data fra egendefinerte kunnskapsbaser, se figur 10 (Martineau, 2024). RAG-rammeverket brukes derfor hovedsakelig grunnet at AI, til tross for sine bemerkelsesverdige evner til å simulere menneskelig intelligens og enorme mengder parametere, har sine begrensninger når det gjelder kunnskap utenfor dens egen treningsdata. Dette er fordi AI-systemer er avhengige av de dataene de har blitt trent på for å forstå og generere svar (Martineau, 2024). Deres kunnskap er derfor begrenset til mønstrene, eksemplene og informasjonen som var tilgjengelig i treningsdatasettet. RAG var derfor essensiell for å utvide AI-assistenten sin kapasitet til å gi nøyaktig og oppdatert veiledning knyttet til FS, selv på forespørsler som strekker seg utover dens opprinnelige treningsomfang.



Figur 10: RAG hovedkomponenter

Etter kontinuerlig testing på UiA sin datamaskin forstod vi at risikoen vi hadde forutsett ble til en realitet. Maskinvaren var ikke tilstrekkelig og testingen av fine-tuning måtte stanses fullstendig ettersom vi ikke hadde verken ressurser eller tid til gode som vi kunne bruke til å fortsette utviklingen med denne teknikken. Derfor tok gruppen et endelig valg om å integrere RAG i vår AI-assistent. Det var på dette punktet fokuset ble fullstendig skiftet over til den mindre maskinvarekrevende teknikken.

Den eksterne kilden som ble brukt for RAG systemet i dette prosjektet var kunnskapsbasen levert av arbeidsgiver. Dette var, som nevnt tidligere, en stor PDF på 410 sider og 50 "mock"-saker som simulerte veiledningssaker med ulike temaer og problemer. Problemet med dette er at et RAG er svært avhengig av datakvalitet for å kunne gi gode svar. Denne PDF-kunnskapsbasen inneholdt mye semistrukturert data i form av tabeller, noe som skaper utfordringer for konvensjonelle RAG-systemer på grunn av to hovedårsaker. Den første årsaken er at det oppstår problemer under tekst-delingsprosessen da tabeller kan bli utilsiktet skilt, noe som fører til data-korrupsjon under uthenting. Det andre problemet oppstår fordi det å inkludere tabeller i dataen kan komplisere semantiske likhets søk (Gao et al., s. 7). For å bekjempe dette tok gruppen i bruk den tidligere prosesserte kunnskapsbasen med ren tekst under testingen, for å unngå problemer forårsaket av strukturen på dataen.

Testing ble hovedsakelig gjort gjennom Jupyter notebook. Her brukte gruppen med den minste modellen tilgjengelig fra Meta sine Llama 2 modeller lokalt på en maskin som tilhører et gruppemedlem. Dette var mulig ettersom RAG ikke krever trening av en AI modell slik som fine-tuning, som betyr at man kan utnytte svakere maskinvare. Og ikke lenge etter avviklingen av fine-tuning utførte gruppen det første vellykkede forsøket på å få svar fra AI modellen med RAG (vedlegg 4).

Etter det første vellykkede forsøket fortsatte gruppen å teste systemet for å forsikre seg om at det var robust nok til å fungere som forventet, uten at det oppstod store avvik. Alt fungerte som det skulle og dermed rettet gruppen nå fokuset videre mot å migrere systemet fra utviklingsmiljøet i Jupyter Notebook, til å få det over til en komplett python-fil.

3.6.3 Sprint review

Siden gruppen i denne sprinten fikk sitt første vellykkede forsøk på å få svar fra AI ved bruk av RAG, var dette et stort vendepunkt for prosjektet, da vi nå hadde oppnådd et konseptuelt bevis på at det var mulig å gi veiledning til FS-systemet gjennom bruken av kunstig intelligens. Dermed oppnådde vi det abstrakte målet som ble satt i sprint planningen til Sprint 5 som var "å svare på om det fortsatt var mulig å lage en lokal open-source AI-assistent for FS, om den begrensede mengden med ressurser som gruppen hadde tilgang på førte til en stopp i fremgangen på fine-tuning". Under sprintens statusmøte uttrykte både oppdragsgiver og veileder at dette var et stort positivt steg i riktig retning for prosjektets mål.

3.6.4 Sprint retrospective

Til tross for det positive vendepunktet i prosjektet, var det fortsatt mye som kunne gjøres for å forbedre AI-assistenten og brukeropplevelsen til veiledningsprosessen. Som sagt tidligere så var all testing på dette punktet gjort i et Jupyter Notebook utviklingsmiljø, og oppdragsgiver oppfordret til å ikke legge mye vekt på brukergrensesnittet, men heller fokusere på det tekniske. Dermed ble det bestemt at brukergrensesnittet skulle bestå av noe lett og brukbart.

Endringen av systemarkitekturen fra Sprint 3 til nåværende versjon har vært betydelig (se figur 11). Som forklart tidligere i sprinten endret vi fra en fine-tunet AI-modell til RAG (Retrieval-Augmented Generation). Denne endringen ble gjort for å håndtere begrensningene vi opplevde med maskinvare for fine-tuning og for å forbedre systemets evne til å gi nøyaktige og relevante svar.

Forklaring av den nye systemarkitekturen:

- Brukergrensesnitt (Grensesnitt for Bruker):
    - Brukeren starter med å sende en forespørsel gjennom brukergrensesnittet.

- Indexing og Vektor-database:
    - Forespørselen blir sendt til indeksen i vektor-databasen.
    - Kunnskapsbasen, som inneholder brukerveiledninger og relevant informasjon, er først behandlet og indeksert. Dokumenter blir segmentert og omgjort til

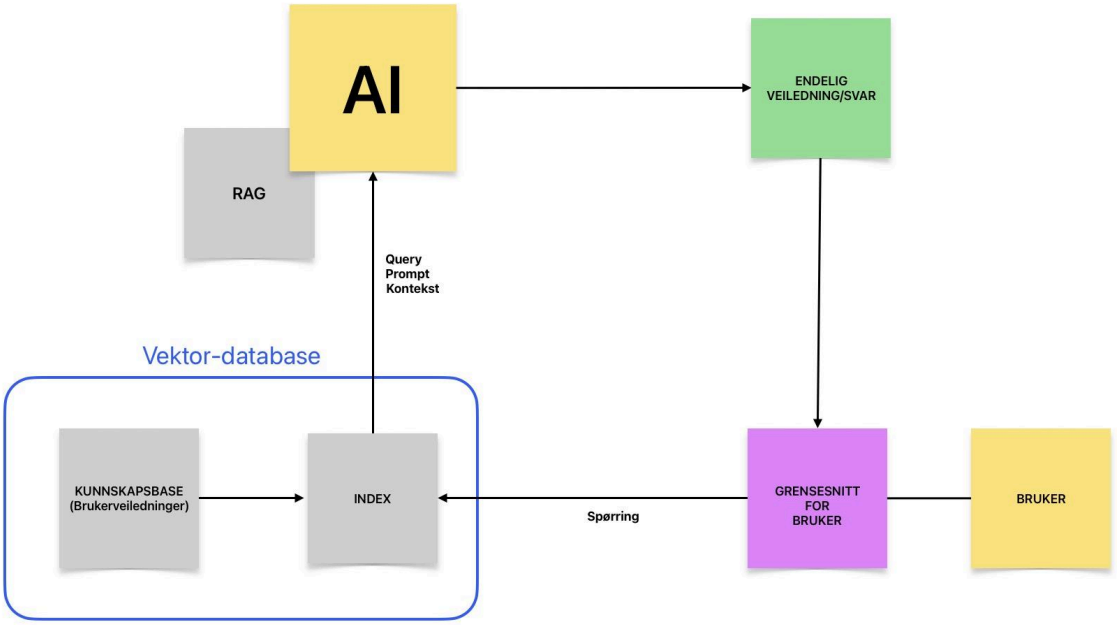
embeddings som lagres i vektor-databasen. Indeksen brukes for å raskt hente relevant informasjon basert på brukerens forespørsel.

- RAG (Retrieval-Augmented Generation):
    - RAG-modulen mottar forespørselen sammen med kontekst fra indeksen. RAG henter den relevante informasjonen fra vektor-databasen for å gi AI-modellen den nødvendige konteksten.
    - Ved å bruke denne konteksten kan AI-modellen generere et svar som er både relevant og nøyaktig. RAG sørger for at AI-assistenten har tilgang til oppdatert og spesifikk informasjon fra kunnskapsbasen.

- AI-modell:
    - AI-modellen mottar query, prompt og kontekst fra RAG.
    - AI-en bruker denne informasjonen til å generere det endelige svaret eller veiledningen.

- Endelig veiledning/svar:
    - AI-assistenten leverer det genererte svaret tilbake til brukergrensesnittet, som presenterer det for brukeren.
    - Denne nye arkitekturen med RAG og vektor-database har gjort systemet vårt mer robust og effektivt. Den håndterer de tidligere utfordringene med maskinvarebegrensninger og forbedrer AI-assistentens evne til å gi nøyaktige og relevante svar. Overgangen til RAG har også gjort det mulig å integrere semistrukturert data fra kunnskapsbasen, noe som ytterligere forbedrer kvaliteten på informasjonen som hentes og brukes.

Totalt sett har denne endringen vært avgjørende for prosjektets fremgang, og den gir oss et sterkt fundament for videre utvikling og forbedring av AI-assistenten. Dette viser at vi nå har en skalerbar løsning som kan håndtere komplekse forespørsler med større nøyaktighet og effektivitet.



Figur 11: Systemarkitektur versjon 4

3.7 Sprint 6

(08.04 - 19.04)

Sprint 6 skulle handle om å sette opp en prototype av AI-assistenten for å dermed teste den.

Verktøy og rammeverk for denne sprinten:

- Gradio
- Llama Index
- Pinecone
- Streamlit
- RunPod
- GPT-3.5 Turbo

3.7.1 Sprint planning

Fokuset for Sprint 6 var å fullføre en fungerende prototype slik at den kunne vises fram. Ut fra forrige statusmøtet med oppdragsgiver kom det formeninger om at denne prototypen ikke var ment å ha alle funksjonene fra gruppens wireframe, men den måtte ha en funksjonerende AI som brukeren kunne stille spørsmål til og få svar. I tillegg var det sentralt at brukeren kunne ha en sammenhengende samtale.

For å nå målet med denne sprinten måtte programmets kode formateres fra et notebook-oppsett til et mer komplett Python-program. Det var også nødvendig å implementere et enkelt brukergrensesnitt, og gruppen utforsket derfor ulike hjelpemidler og maler spesifikt utviklet for chatbot-løsninger. Dette ble gjort fordi gruppen ikke hadde mulighet til å bruke tid på å bygge en frontend fra bunnen av, og målet var å få en velfungerende chatbot.

3.7.2 Sprint gjennomføring

Det første relevante brukergrensesnittsverktøyet var **Gradio**, siden den er utviklet av Hugging Face. Dette betyr at den har direkte integrasjon med Spaces som gjør det lett tilgjengelig å publisere programmet slik at den kan deles. Spaces på Hugging Face tilbyr prosessorkraft, slik at dersom gruppen velger å ta i bruk en krevende språkmodell, kan dette gjøres på Spaces. Gratisversjonen av Gradio tilbyr HTTP forespørsler uten en dedikert server, noe som gjør det mulig for gruppen å dele applikasjonen umiddelbart via en enkel lenke (Gradio, u.å.).

Gradio har utviklet flere testkoder som kan kjøres på Google Colab. Disse er åpne og tilgjengelige for alle å teste ut. Gruppen brukte disse kodesnippene til å tilpasse etter gruppens behov og konfigurasjon. Ut fra disse testene kunne gruppen finne hvilket kodesnipp som var relevant og kompatibelt med RAG. På grunn av det enkle brukergrensesnittet til Gradio, er mulighetene til konfigurasjon og tilpasning små. Gradio har blant annet ikke funksjonaliteten til å ta hensyn til brukerens samtale, noe som andre brukergrensesnittsverktøy har metoder for. Derfor valgte gruppen å utforske andre verktøy.

For å oppnå muligheten til at systemet kan innhente og ta hensyn til brukerens samtale, kan datarammeverk som **Llama Index** med innebygd funksjonalitet brukes som en løsning. Llama Index Chat lar systemet og språkmodellen generere tekst og holde en samtale ut fra egendefinerte dokumenter, som PDF-filer. Llama Index har mulighet til å anvende RAG-teknikken, som nevnt i forrige sprint. En fordel er at Llama Index håndterer data indeksering av numeriske vektorene som blir generert fra dokumentene lokalt. Dette gjør at systemet blir fullstendig lukket, og dermed mister behovet for en nettbasert vektordatabase som Pinecone (Awan, 2023). Dersom språkmodellen og kunnskapsbasen befinner seg lokalt på kjøringen kan dette systemet brukes, selv når internettilkobling ikke er tilgjengelig.

Et alternativ for brukergrensesnittsverktøy er **Streamlit**. Dette er en Python-pakke med åpen kildekode som enkelt kan installeres ved bruk av kommandolinjen. Streamlit tilbyr flere dynamiske funksjoner og metoder som kan benyttes av AI og maskinlæring. Noen av de sentrale metodene i Streamlit er for å huske brukersamtalen og generere tekstfelter for brukerinnputt. Det er disse metodene prosjektet er bygget på og benytter seg av. Denne metoden for å lage et brukergrensesnitt gjennom Streamlit kan oppnås ved bruk av færre kodelinjer enn dersom koden skulle skrives fra bunnen av (Streamlit, u.å.). Disse funksjonene

har fokus på brukergrensesnittet, og derfor mindre teknisk nyttig. Som nevnt tidligere, tilbyr Gradio umiddelbar deling. Dersom programmet skal kjøres med Streamlit og bli delt, må kjøringen kjøres med en host. Denne kan være fremmet på egne servere eller ved bruk av nettbaserte delingsverktøy som LocalTunnel. Dette er en enkel måte å dele en lokal kjøring midlertidig, og det må brukes dersom Streamlit blir kjørt gjennom Google Colab.

Svarene fra Llama 2 var ikke tilstrekkelige til å bli publisert og brukt i et virkelig tilfelle. Dette var på grunn av tiden det tok for Llama 2 å generere et svar basert på kunnskapsbasen var betraktelig høy med tid på maskinvaren den ble kjørt på. For å sette det i perspektiv, kunne en Llama 2 spørring bruke opp til tre minutter på å generere et svar. Begrensningene i maskinvare kan overkommes ved bruk av nettbaserte tjenester som tilbyr lån av maskinkraft. **RunPod** er spesifikt laget for å trene og kjøre AI og maskinlæring, og derfor er det høy kompatibilitet med programmeringsspråket Python som er brukt for prosjektet. Dette gjorde overgangen til å kjøre koden på RunPod sømløs. RunPod tilbyr flere typer maskinvarer til forskjellige behov (RunPod, u.å.).

I denne fasen skulle RunPod bare brukes til testing, og derfor hadde prosjektet ingen spesifikke behov for de mest moderne maskinvarene. Denne overgangen til skybaserte maskinvare åpnet prosjektet opp for mulighet til testing på større språkmodeller som har anledning til å svare bedre enn Llama 2 med 7 milliarder parametre. Den største og mest krevende Llama 2 modellen er versjonen med 70 milliarder parametre. Det er denne som ble testet ut på RunPod, men siden denne versjonen av Llama 2 ikke var godt fine-tuned på norsk data, kunne den ikke svare på norsk. Derfor ble denne løsningen irrelevant for prosjektet.

For å både få et bredere og tidsbesparende svar på norsk, ble koden formatert slik at det ble mulig å bruke GPT-modeller via API fra OpenAI som har utgitt ulike betalte API-versjoner som skaper et grensesnitt, slik at GPT-modellene kan bli brukt i andre applikasjoner. GPT står for *Generative Pre-trained Transformer*, som betyr at denne modellen tilhører transformator-arkitekturen. Denne arkitekturen er laget for å prosessere naturlig språk (Ipsen, 2023). Etter denne rapporten blir publisert, er det for øyeblikket den mest oppdaterte versjonen fra Openai GPT-4 Turbo. Denne versjonen klarer å svare på komplekse oppgaver, men med tanke på at den er større sammenlignet med tidligere versjoner, er den også tregere. Antallet parametre for GPT-4 Turbo er ikke kjent, siden OpenAI ikke har offentliggjort denne informasjonen. På grunn av budsjettbegrensninger har gruppen valgt å bruke en tidligere versjon, **GPT-3.5 Turbo**. Denne består av 175 milliarder parametre, og krever derfor store datasentre å kjøre (Raghunath, 2023).

3.7.3 Sprint review

Sjette Sprint review ble gjennomført med oppdragsgiver og veileder. Gruppen presenterte en fungerende prototype av AI-assistenten til oppdragsgiver og veileder. Prototypen hadde

et enkelt brukergrensesnitt utviklet ved hjelp av Streamlit, og den kunne håndtere sammenhengende samtaler ved å benytte Llama Index. Testing ble utført med både Llama 2 ved bruk av RunPod og gjennom OpenAI sin API nøkkel.

Oppdragsgiver og veileder var tilfredse med fremgangen og funksjonaliteten til prototypen, spesielt med tanke på at den kunne gi relevante svar på brukernes spørsmål. Overgangen fra Llama 2 til GPT-3.5 Turbo resulterte i raskere og mer relevante svar, noe som ble godt mottatt. Det ble også diskutert at budsjettet fremover vil inkludere kostnader for bruk av GPT-3.5 Turbo og eventuelt GPT-4 Turbo dersom det er økonomisk forsvarlig. På bakgrunn av dette konkluderte gruppen med at målene satt i sprint planningen ble oppnådd.

3.7.4 Sprint retrospective

Etter endringen fra Llama 2 til GPT-3.5 Turbo ble de genererte svarene raskere og mer relevante mot kunnskapsbasen. Kostnadene ved bruk av GPT-3.5 Turbo ble finansiert av gruppemedlemmene, men kom til enighet med oppdragsgiver på statusmøtet at prosjektet skal få et budsjett. Dette åpnet opp tilgang til GPT-4 Turbo, som har mulighet til å svare basert på mer data, men koster for øyeblikket 20 ganger mer enn GPT-3.5 Turbo. Dermed er det viktig å bare bruke GPT-4 Turbo dersom budsjettet tillater dette. Til tross for at endringen fra Llama 2 til GPT ble utført, bestemte veileder og oppdragsgiver å fremdeles opprettholde Llama 2-versjonen.

Overføringen fra et notebook-opplegg til et mer komplett Python-program viste seg å være mer krevende enn det som ble forestilt. Dette var delvis på grunn av oppsettet til notebooks. Fordelen med disse går ut på at de blander både tekst og kjørbar kode i samme fil, noe som tillater notering og forklaring der koden er. Denne fordelen blir til en ulempe når notebook-filen skal bli gjort om til et velfungerende program med tanke på at den får problemer med å skille hva som er notater og kjørbar kode. I tillegg blir Python-klasser ikke alltid konvertert riktig over, og kommer ofte i feil rekkefølge. Det er disse Python-klassene som tar tid å manuelt konvertere.

3.8 Sprint 7

(22.04 - 03.05)

Sprint 7 ble dedikert til å justere svarene fra språkmodellen ved bruk av system-prompt.

Verktøy og rammeverk for denne sprinten:

- System-prompt
- Datavask i Microsoft Office Excel

3.8.1 Sprint planning

Til tross for at AI-en har gitt svar basert på kunnskapsbasen, er det fremdeles rom for forbedringer når det gjelder henvisning til kunnskapsartikler. Under brukertesten utført i Sprint 3 kom det fram at intervjuobjektet ønsket at lenken til nettsiden for den spesifikke veiledningen som ble gitt av AI-assistenten også skulle inkluderes i svaret. Derfor var det essensielt at AI-assistenten hadde mulighet til å finne riktig lenke til den respektive kunnskapsartikkelen. Målet for denne sprinten var derfor å implementere en løsning som effektivt lar AI-en hente riktig veiledning og lenke fra kunnskapsbasen.

3.8.2 Sprint gjennomføring

Gruppen startet med å implementere en **system-prompt** gjennom prompt engineering, som skulle forbedre nøyaktigheten og relevansen av svarene AI-assistenten gir. Prompt engineering er en teknikk der man gir spesifikke instruksjoner på hvordan AI-en skal oppføre seg (IBM, u.å.). Denne system-prompten ble designet for å gi AI-en kontekst om hva den skal hjelpe med og hvordan, i tillegg til å inkludere spesifikke lenker fra kunnskapsbasen sammen med selve veiledningen. System-prompten ble gjennom resten av prosjektløpet iterert flere ganger for å teste hvordan endringene påvirket svarene som ble gitt. Den mest oppdaterte system-prompten vil være inkludert i kodebasen på GitHub.

For at AI-en best mulig skal kunne hente ut riktig lenke, ble det foreslått av veileder å integrere lenkene inn i midten av alle kunnskapsartiklene. Alle kunnskapsartikler er delt opp slik at de har et kort sammendrag og selve kunnskapsartikkelen. Dette gjør at lenkene kan bli satt inn mellom disse to elementene. For å gjøre dette på en enkel måte, ble dataprogrammet **Microsoft Office Excel** brukt for å utføre dette forslaget, og i tillegg vaske kunnskapsbasen for eventuelle blanke og tomme felter. Dette verktøyet tilbyr mange ulike funksjoner som er relevante for å utføre denne formateringen.

3.8.3 Sprint review

Syvende Sprint review ble gjennomført med oppdragsgiver og veileder. I begynnelsen av Sprint 7 hadde gruppen som mål å få AI-en til å hente ut riktig veiledning og lenke til veiledningen som ble gitt. Siden AI-en klarte dette på en effektiv og strukturert måte, ble målet for Sprint 7 oppnådd

3.8.4 Sprint retrospective

Fra de testene gruppen utførte, viste det at AI-en kunne hente ut riktig hyperkobling oftere ved å formatere kunnskapsbasen på den måten veileder foreslo. Selv om formateringen tok betraktelig med tid, viste resultatene at svarene inneholdt ekte hyperkoblinger som har blitt hentet fra kunnskapsbasen. Det er fremdeles tilfeller der hyperkoblingene er fullstendig oppdiktet, og går dermed ikke til en ekte kunnskapsartikkel. Derfor er det fortsatt endringer i

systemet som kan bidra til å minske tilfellene der AI-en svarer med ingen eller feil hyperkobling.

3.9 Sprint 8

(06.05 - 16.05)

Sprint 8 var den siste og endelige sprinten i prosjektet, der fokuset var å fullføre dokumentasjonen av hele prosessen og å fortsette med forbedring av systemet.

Verktøy og rammeverk for denne sprinten:

- Evalueringstest

3.9.1 Sprint planning

Hovedmålet med Sprint 8 var å fullføre den endelige rapporten og dokumentasjonen av prosjektets livssyklus, samtidig som gruppen fortsatte å iterativt forbedre systemet på små områder der det var mulig. Dette skulle bli gjort gjennom evaluering av "chunk sizes" og en ny brukertest for å få tilbakemelding på systemets ytelse og tilstand.

3.9.2 Sprint gjennomføring

Som forklart i “Sprint planning”, gikk gjennomføringen av denne sprinten hovedsakelig til å ferdigstille dokumentasjonen av prosessene som gruppen gikk gjennom i dette prosjektet. For at prosjektet skulle være mulig å etterprøve var det viktig å oppdatere den eldre dokumentasjonen, samt føre opp gruppens siste oppsett av den nyeste versjonen av produktet. Parallelt med dokumentasjonen hadde gruppen tid til mer fremgang i forbedringen av hvordan AI-assistenten svarer.

Videre forsøkte gruppen å forbedre systemet med minimale endringer ved å justere "chunk sizes". I indekseringsfasen blir dokumenter behandlet, segmentert og omgjort til embeddings som lagres i en vektordatabase. Kvaliteten på denne indekseringsprosessen er avgjørende for om riktig kontekst kan hentes i senere faser. Den vanligste metoden er å dele dokumentet opp i “chunks” på et fast antall tokens (f.eks. 256, 512, 1024). Større "chunks" kan fange mer kontekst, men de genererer også mer støy, krever lengre behandlingstid og høyere kostnader. Mindre "chunks" gir mindre støy, men kan ikke alltid formidle nødvendig kontekst fullt ut (Gao et al., s. 8). Derfor er det viktig å finne den riktige balansen på størrelsen av "chunks" for å optimalisere ytelsen.

For å måle denne ytelsen utførte gruppen en **evalueringstest** ved hjelp av LlamaIndex sine evalueringsmoduler, som måler kvaliteten på systemets uthenting. De to modulene som ble brukt i denne testen er “troskaphet” og “uthentings-evaluering”. “Troskaphet” måler om svaret til AI-assistenten er sann til den uthentede konteksten (dvs. om det oppstår

hallusinasjoner fra AI-en eller ikke), mens “utehentings-evalueringsmodulen” måler om kildene som er brukt, er relevant til brukerens spørring. Denne testen bruker en stor språkmodell til å bestemme om det predikerte svaret er riktig målt opp mot spørsmål som er generert fra kunnskapsbasen (Evaluating - LlamaIndex, u.å.). For vårt system bestemte gruppen seg for å teste “chunk” størrelsene 512, 1024 og 2048, resultatene fra testen var som følger:

For chunk-størrelse 512

- Gjennomsnittlig responstid: 2.89 sekunder
- Gjennomsnittlig Troverdighet: 0.75
- Gjennomsnittlig Relevans: 0.75

For chunk-størrelse 1024

- Gjennomsnittlig responstid: 2.65 sekunder
- Gjennomsnittlig Troverdighet: 0.95
- Gjennomsnittlig Relevans: 0.85

For chunk-størrelse 2048

- Gjennomsnittlig responstid: 4.62 sekunder
- Gjennomsnittlig Troverdighet: 0.55
- Gjennomsnittlig Relevans: 0.45

Resultatene viste at en chunk-størrelse på 1024 var den mest optimale, med den beste balansen mellom responstid, troverdighet og relevans. Større chunks (2048) økte responstiden betydelig og reduserte både troverdighet og relevans, mens mindre chunks (512) hadde raskere responstid, men ikke samme nivå av troverdighet og relevans som de på 1024 tokens. Dette indikerer at en mellomstor chunk-størrelse er mest effektiv for å sikre både rask og nøyaktig respons fra AI-assistenten.

Mot slutten av Sprint 8 ble det også utført brukertesting for å evaluere systemets funksjonalitet og brukervennlighet. Brukertesten bestod av 15 klargjorte scenarioer som vi ønsket at brukeren skulle bruke til å stille spørsmål til AI-assistenten. Disse scenarioene var utformet for å dekke et bredt spekter av potensielle brukssituasjoner og sikre at systemet kunne håndtere ulike typer forespørsler. I tillegg stilte brukeren noen spørsmål utenfor disse scenariene fra egen erfaring, noe som ga oss verdifull innsikt i hvordan AI-assistenten håndterer uforutsette spørsmål og tilpasser seg ulike bruksmønstre.

3.9.3 Sprint review

I denne Sprint reviewen var dessverre ikke oppdragsgiver og veileder inkludert for å hjelpe til med evalueringen, men gruppen prøvde da å anvende erfaringen fra de tidligere tilbakemeldingene til å reflektere resultatene til vår beste evne. Hovedmålet med å fullføre dokumentasjonen til prosjektet på dette tidspunktet var estimert til å bli oppnådd, samt at gruppen kunne etter evaluerings-testen som ble utført ta en informert beslutning som kunne bidra til å forbedre systemet. Brukertesten ga oss også nyttig tilbakemelding som kan tas med videre. Brukeren uttrykte at man nå kunne se at brukerstøtten potensielt ville spare tid og var effektivisert, selv om den likevel ikke ga perfekte svar. Denne tilbakemeldingen inkluderte også innsikt i hvordan AI-assistenten håndterer både de klargjorte scenarioene basert på kunnskapsbasen og uforutsette spørsmål, noe som viste at systemet har en god evne til å tilpasse seg ulike bruksmønstre.

3.9.4 Sprint retrospective

Totalt sett, til tross for manglende direkte evaluering fra oppdragsgiver og veileder, var sprinten vellykket i å nå sine mål. Dokumentasjonen ble ferdigstilt, og både evaluerings- og brukertester ga verdifulle innsikter for videre forbedringer.

4. Evaluering

I denne delen av rapporten evaluerer vi prosjektets gjennomføring, resultater og oppnådde mål. Vi presenterer en konklusjon basert på vår erfaring og innsikt, samt hvordan vi har overvunnet ulike utfordringer. Videre gir vi anvisning for etterprøving av produktet, inkludert en lenke til vårt GitHub-repository som inneholder all nødvendig dokumentasjon og instruksjoner for testing.

4.1 Konklusjon

Prosjektet var en utprøving av konsept med hensikt å bruke kunstig intelligens til å effektivisere brukerstøtte til Felles studentsystem på UiA ved å basere den på egen data. Hensikten var å vurdere AI-ens evne til enten å svare på henvendelser autonomt eller å fungere som støtte for en saksbehandler.

I betraktning av prosjektets natur som en konseptutprøving, kombinert med uklarheten i prosjektet og gruppens begrensede forhåndskunnskaper om det aktuelle domenet, var det ikke forventet å komme frem til en perfekt løsning. Målet var snarere å utvikle en tilnærming som kunne testes og vurderes, slik det detaljert er beskrevet i avsnitt 4.1.1, "Til etterprøving".

Scrum rammeverket, spesielt sprintene, var grunnleggende for gruppen hvor prosjektplanen ikke var nedfelt i minste detalj, men heller overfladisk. Dette la grunnlaget for en betydelig del av studentenes arbeid med å navigere og tilpasse seg disse hindringene etter hvert som de oppstod.

Gruppen har kontinuerlig hatt kommunikasjon med oppdragsgiver gjennom ukentlige statusmøter, som gruppen selv har både arrangert og ledet.

Totalt sett kan man si at prosjektet er vellykket og kan bygges videre på.

4.1.1 Til etterprøving

For å etterprøve og teste det nåværende produktet, har gruppen opprettet et GitHub-repository som inneholder all nødvendig kode og instruksjoner. Repositoriet er delt inn i tre mapper, hver av dem med en egen README.md-fil som gir detaljerte veiledninger for hvordan man kan sette opp og kjøre systemet selv.

For å teste systemet lokalt, følg instruksjonene i README.md-filene i de respektive mappene i GitHub-repositoriet. Disse instruksjonene gir en steg-for-steg guide til hvordan man setter opp miljøet, laster ned nødvendige avhengigheter, og kjører applikasjonen.

GitHub-lenke: https://github.com/Edi-G/FS-H

4.2 Prosjekt retrospektiv

Da vi startet prosjektet definerte vi kvalitet som “hvor mye egenskaper eller karakteristikker ved et produkt, tjeneste eller prosess måler seg opp mot behov, krav eller forventninger som er angitt. ” Derfor vil vi i dette kapittelet reflektere over hvorvidt vi har oppnådd dette i prosjektet.

Forventningene innad i gruppen har blant annet vært å innfri oppdragsgiverens krav, behov og forventninger.

Oppdragsgivers krav til prosjektet var følgende:

- Gi opplæring i generelt vedlikehold. Kort gjennomgang av hvordan systemet teknisk er lagd.
- Presentere oppgave for oppdragsgiver i separat møte.
- Statusmøte en gang i uken gjennom prosjektet sin varighet. Arrangeres og holdes av studentene.
- Effektivisere brukerstøtten til Felles system.

Det kan ikke bekreftes av gruppen selv, i hvert fall ikke enda, at man har gitt opplæring i generelt vedlikehold. Likevel har rapporten dokumentert hvordan systemet er bygd opp gjennom analyse, systemarkitekturer, systemdefinisjon, use cases, og til slutt gjennom beskrivelser av systemet kontinuerlig i rapportens løp. Dette vil også bli presentert i det separate møtet som vil bli gjennomført mellom bachelorrapportens leveranse og muntlig eksamen. Dermed vil kravet om å presentere oppgaven for oppdragsgiver bli innfridd i etterkant av rapporten. Kravet om ukentlige statusmøter er også innfridd, og har blitt både arrangert og holdt av gruppen. Basert på uttalelsen fra oppdragsgiver (vedlegg 1) er også oppdragsgiver enige med oss. Kravet om å effektivisere brukerstøtte kan bli innfridd fra resultatet av brukertesten i Sprint 8. Brukertesten ga innsikt i brukbarheten av det utviklede systemet. Selv om AI-assistenten ikke klarte å gi tilfredsstillende brukerstøtte på alle av scenarioene, fikk gruppen bekreftet av tilbakemeldingen fra brukeren at den effektiviserte brukerstøtten.

Oppdragsgivernes forventninger ble satt sammen med gruppen på Sprint reviewene og de ukentlige statusmøtene som ble holdt i løpet av prosjektet. På grunnlag av tilbakemeldinger og erfaringer fra sprintene har vi fått inntrykk av at forventningene har blitt tilfredsstilt.

Oppdragsgivers behov til prosjektet var følgende:

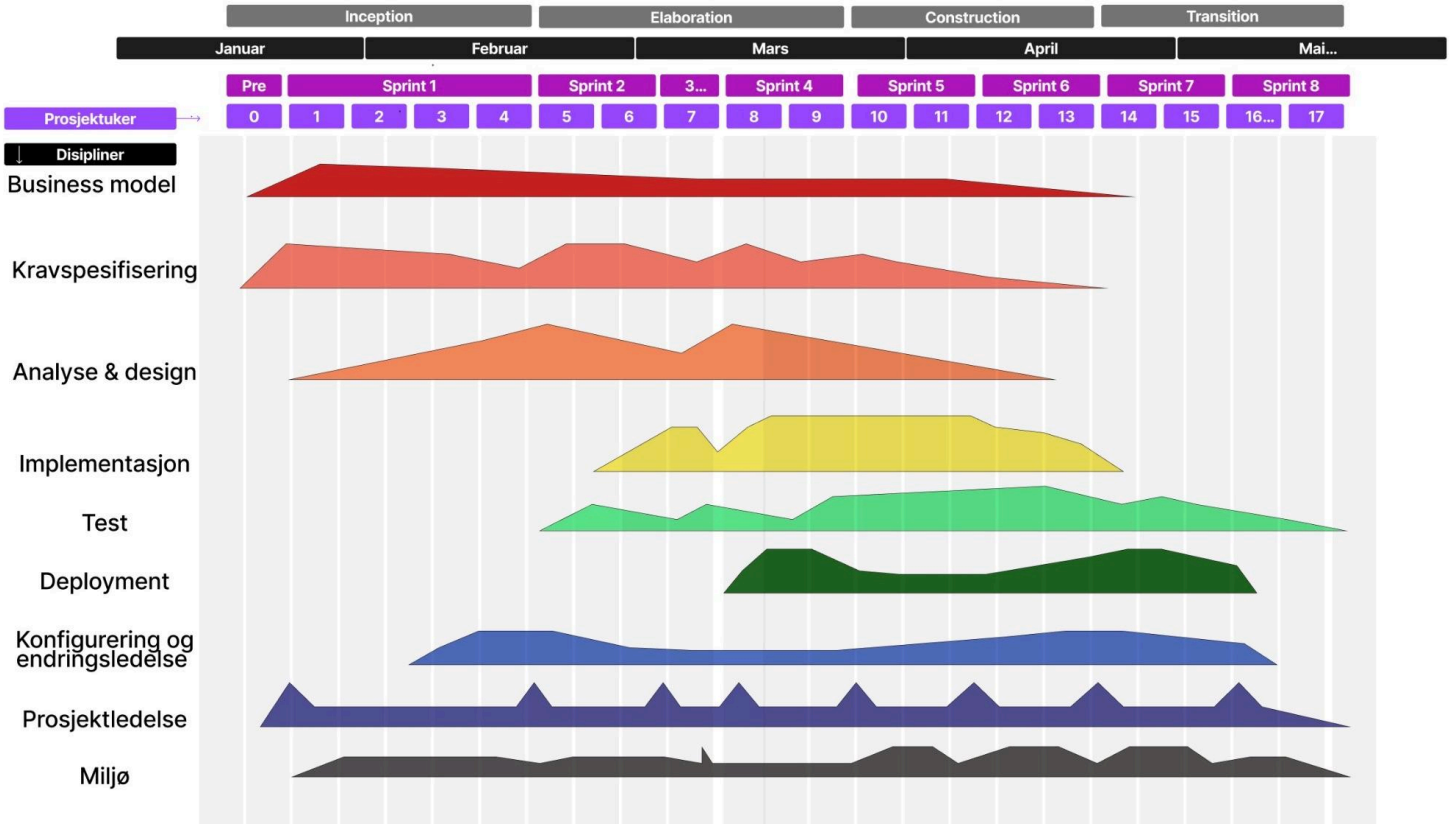
- Vite at gruppen har et tilfredsstillende og godt læringsutbytte, samt følger UiA sine retningslinjer.
- Kontroll over prosjektets fremdrift og fremtidige planer.

I løpet av dette prosjektet har gruppen utforsket og anvendt en rekke teknologier, som er oppsummert i verktøylisten (se vedlegg 3), der mesteparten er metoder og verktøy vi ikke hadde erfaring med fra før prosjektet. Gjennom bruk av disse teknologiene har vi tilegnet oss en dypere forståelse av deres funksjonalitet. Av disse har vi også anskaffet oss en dypere forståelse ved å teste og anvende dem til vårt eget prosjekt. Der er teknologiene som følger: RAG, Fine-Tuning, Llama 2, Llama Index, Chat GPT, Hugging Face, Google Colab og System Prompting. Som en bekreftelse av dette uttrykte oppdragsgiveren i uttalelsen sin, vedlegg 1, at gruppen har tilegnet seg vesentlig kunnskap gjennom prosjektet.
Gruppen har også anvendt og fått erfaring fra metoder innenfor prosjektstyring. Derav hvor vanskelig det er å estimere tid, eller i vårt tilfelle innsats på de diverse brukerhistoriene. Dette er fordi det er to hovedmetoder for estimering: ekspertbasert og datastyrt. Mangelen på ekspertråd har begrenset vår evne til å vurdere størrelse, kompleksitet, og tilgjengelige ressurser på en effektiv måte. I tillegg har vi ikke tidligere erfaring fra lignende prosjekter som kunne hjulpet oss med å danne et mer nøyaktig bilde av innsatsen (Alsaadi & Saeedi, 2024). Som et resultat har vi hatt vanskeligheter med estimering av backlog items, og vi vet nå at man burde utforske lignende prosjekter eller søke råd fra ekspert for bedre estimater.

Med dette kan gruppen si at samtlige har hatt kunnskapsheving i flere områder, både på det tekniske aspektet og ved prosjektyrings aspektet, noe som betyr at gruppen hadde et tilfredsstillende og godt læringsutbyttet underveis i hele prosjektet.

I ettertid av prosjektet har gruppen også erfart at prosjekter aldri går etter den initielle planen. Gruppen bekjenner at det var flere faktorer som ikke gikk etter planen. Siden vi brukte prinsippene til Scrum og *Det Agile Manifestet* som grunnstein for vår prosjektstyring har vi kunnet endret kurs underveis i gjennomføringen. I tillegg til å vise vår evne til å gjennomføre og styre et prosjekt, har vi også lært av erfaringene underveis i dette prosjektet og styrket den evnen.

Andre eksempler på erfaringer fra prosjektstyringen fremstod da den tentative tidsplanen for prosjektet vårt, vedlegg 5, endret retning allerede etter Sprint 2. Det var flere ganger i prosjektløpet dette skjedde, og mye av årsaken til dette var utfordringer som oppstod utenfor vår kontroll. Disse utfordringene var risikoer som vi ikke klarte å forutse da vi utførte en risikoanalyse i Sprint 1, vedlegg 8. Da vi mistet tilgang til ressurser underveis i prosjektet, som med Microsoft Azure, er et eksempel på risikoer vi ikke forventet som ble til utfordringer. Dette gjorde at vi alltid måtte prøve å tilpasse oss etter hvordan den foregående sprinten resulterte, noe som i prinsippet gjorde oss agile. Sammenliknet med den tentative tidsplanen lærte vi også at selv om man har som team satt opp systemutviklingsprosessene sekvensielt, er det områder de går over hverandre.
Dette kan vi retrospektivt se i et “Rational Unified Process” (RUP) diagram vedlagt i figur 12 (Maulani et al., 2021). RUP er en type prosess innenfor System Development Life Cycle (SDLC) og er en måte å styre utviklingen av et programvareprosjekt, men i vårt tilfelle stemmer diagrammet overens med vårt prosjekt i ettertid av gjennomføringen hvor vi ser at de forskjellige prosessene i Y-aksen, “Disipliner”, overlapper hverandre på flere områder av tidslinjen.



Figur 12: RUP diagram

Litteratur

Adams, M. N. (2021, 10. september). Scrum poker for agile projects. Hentet fra https://www.atlassian.com/blog/platform/scrum-poker-for-agile-projects

Adobe Experience Cloud Team (2023, 8. oktober). Popular project management methodologies. Hentet fra https://business.adobe.com/blog/

AlmaBetter. (u.å). *Introduction to Google Collab and Almabetter IDE*. Hentet fra https://www.almabetter.com/bytes/tutorials/python/introduction-to-google-colab-and-almabetter-ide

AI Verse. (2023). *What Is Llama Cpp?* Hentet fra https://aiverseinfo.com/what-is-llama-cpp/

Alsaadi, B., & Saeedi, K. (2024). Ensemble effort estimation for novice agile teams. Information and Software Technology, 170, 107447. Hentet fra https://doi.org/10.1016/j.infsof.2024.107447

Aspelund, M. (2024, January 26). Hva er et API og en API integrasjon? Hentet fra https://www.vismasoftware.no/artikler/hva-er-et-api-og-en-api-integrasjon

Aston, B. (2024, 14. mars). 10 Reasons Why Project Management Is So Important For Orgs. Hentet fra https://thedigitalprojectmanager.com/personal/new-pm/why-is-project-management-important/

Atlassian. (u.å.). Agile. Hentet fra https://www.atlassian.com/agile

Aven, T. (2018, 18. september). 1 Om risiko og usikkerhet. Hentet fra https://www.regjeringen.no/no/dokumenter/nou-2018-17/id2622043/?ch=6

Aven, T. (2022, 10. oktober). Risikomatrise. Hentet fra https://snl.no/risikomatrise

Awan, A. (2023). *LlamaIndex: Adding Personal Data to LLMs*. Hentet fra https://www.datacamp.com/tutorial/llama-index-adding-personal-data-to-llms

Azure OpenAI Service - Pricing | Microsoft Azure. (u.å.). Microsoft Azure. Hentet fra https://azure.microsoft.com/en-us/pricing/details/cognitive-services/openai-service/

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Benyon, D. (2019). Designing user experience. (4. utg.). Pearson Education Limited

Ipsen, A. (2023). GPT Base, GPT-3.5 Turbo & GPT-4: What's the difference? Hentet fra https://www.pluralsight.com/resources/blog/data/ai-gpt-models-differences

Brush, K., & Silverthorne, V. (2022, november). Agile Software Development. Hentet fra https://www.techtarget.com/searchsoftwarequality/definition/agile-software-development

DSPy. (u.å). *About DSPy*. Hentet fra https://dspy-docs.vercel.app/docs/intro

Evaluating - LlamaIndex. (u.å.). Hentet fra https://docs.llamaindex.ai/en/stable/module_guides/evaluating/

Felles studentsystem. (u.å.). Hva er FS? Hentet fra https://www.fellesstudentsystem.no/

Ferrer, J. (2023). *What is Hugging Face? The AI Community's Open-Source Oasis*. Hentet fra https://www.datacamp.com/tutorial/what-is-hugging-face

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., & Wang, H. (2023, 18. desember). Retrieval-Augmented Generation for Large Language Models: A survey. arXiv.org. Hentet fra https://arxiv.org/abs/2312.10997

Google. (u.å.). *Colaboratory: Frequently Asked Questions*. Hentet fra https://research.google.com/colaboratory/faq.html

Gradio. (u.å.). *Using Hugging Face Integrations*. Hentet fra https://www.gradio.app/guides/using-hugging-face-integrations

Gundersen, D., & Halbo, L. (2018, 28. mai). Kvalitet. Store Norske Leksikon. Hentet fra https://snl.no/kvalitet

IBM. (u.å.a). What is Artificial Intelligence (AI)? Hentet fra: https://www.ibm.com/topics/artificial-intelligence

IBM. (u.å.b). *What are LLMs?* Hentet fra https://www.ibm.com/topics/large-language-models.

Lehne, E. & Nisar, B. (2020). Evaluering av Felles studentsystem - Gartner Group. Hentet fra https://www.fellesstudentsystem.no/samarbeid-og-medvirkning/tidligere-samarbeidsmodeller/plangruppemoter/2020/2020-06-10-planleggingsgruppemote/sak_35_20_vedlegg_35a_evaluering_av_felles_studentsystem_-_sluttrapport-.pdf

Martineau, K. (2024, 1. mai). What is retrieval-augmented generation? IBM Research Blog. Hentet fra https://research.ibm.com/blog/retrieval-augmented-generation-RAG

Mathiassen, L., Munk-Madsen, A., Nielsen, P. A., & Stage, J. (2018). Object Oriented Analysis & Design. Hadsund: Metodica ApS.

Maulani, G. a. F., Hamdani, N. A., Bhakti, D. D., & Denni, I. (2021). The management application design of digital archiving letters. IOP Conference Series. Materials Science and Engineering, 1098(4), 042005. Hentet fra https://doi.org/10.1088/1757-899x/1098/4/042005

Meta. (2023). *Meta and Microsoft Introduce the Next Generation of Llama*. Hentet fra https://about.fb.com/news/2023/07/llama-2/

Mikalsen, M. (2024, 22. mars) Brukerhistorie. Store Norske Leksikon. Hentet fra https://snl.no/brukerhistorie

Radigan, D. (u.å.) Story points and estimation. Atlassian. Hentet fra https://www.atlassian.com/agile/project-management/estimation#:~:text=Story%20points%20are%20units%20of,any%20other%20piece%20of%20work.

Raghunath, A. (2023). *GPT-4 Parameters Explained*. Hentet fra https://hix.ai/hub/chatgpt/gpt-4-parameters

Regjeringen (2016). 4.1.3 Tilnærminger til risikovurdering. Hentet fra https://www.regjeringen.no/no/dokumenter/nou-2016-19/id2515424/sec3?q=nou%202016:19

RunPod. (u.å). *Overview*. Hentet fra https://docs.runpod.io/overview

Rutledge, P., & Hogg, J. L. (2020). In-Depth interviews. The International Encyclopedia of Media Psychology. Hentet fra https://www.researchgate.net/publication/345737833_In-Depth_Interviews

Ryles, G. (2022). *What are CUDA Cores?* https://www.trustedreviews.com/explainer/what-are-cuda-cores-4226433

Sander, K. (2023). Dybdeintervju - enkelt intervju. eStudie.no. https://estudie.no/dybdeintervju-enkelt-intervju/

Schwaber, K., Sutherland, J. (2020). The 2020 Scrum Guide. Hentet fra https://scrumguides.org/scrum-guide.html

Schwaber, K., Sutherland, J., & Thomas, D. (2001). Manifesto for Agile Software Development. Agile Alliance. Hentet fra https://agilemanifesto.org/

Scrum.org (u.å.). What is a Sprint Retrospective? Hentet fra https://www.scrum.org/resources/what-is-a-sprint-retrospective

Streamlit. (u.å.). *Streamlit documentation*. https://docs.streamlit.io/

Usability testing: Evaluative UX research methods. (u.å.). https://www.userinterviews.com/ux-research-field-guide-chapter/qualitative-usability-testing

Walker, S. (u.å.) *What is supervised fine-tuning?* https://klu.ai/glossary/supervised-fine-tuning

What is a Burndown Chart? - ServiceNow. (u.å.). ServiceNow. Hentet 19. februar 2024 fra: https://www.servicenow.com/products/strategic-portfolio-management/what-is-a-burndown-chart.html

What is Fine-Tuning? | IBM. (u.å.). https://www.ibm.com/topics/fine-tuning

What is Prompt Engineering? | IBM. (u.å.). https://www.ibm.com/topics/prompt-engineering

What is Python? Executive Summary. (u.å.). Python Software Foundation - Python.org. https://www.python.org/doc/essays/blurb/

Vedlegg

Vedlegg 1: Uttalelse fra oppdragsgiver

Bacheloroppgave våren 2024 for IT og informasjonssystemer

Universitetet i Agder (UIA) har en fremoverlent og utforskende politikk når det kommer til bruk av kunstig intelligens (KI). UiA var på tidspunktet oppgaven ble gitt allerede godt i gang med flere prosjekter innenfor KI. Spesielt var det stor interesse for hvordan KI kunne brukes innenfor undervisning og utdanning. Oppgaven til studentene gikk derimot å rette blikket mot administrasjonen for å se om KI kunne gi noen effektiviseringsløft. Det var allerede startet opp prosjekter rundt bruk av Chat GPT og hvordan slik KI kan brukes til å hjelpe å besvare henvendelser fra studenter og ansatte. For at Chat GPT skal kunne svare må derimot informasjonen finnes offentlig og ligge tilgjengelig. Oppgaven til bachelorstudentene var å se på mulighetene for å bruke andre løsninger hvor det var mulig og trene KI på egne data, slik som veiledninger og tidligere saker. Dette med mål i å se om en KI kan besvare saker enten automatisk, eller ved å assistere en saksbehandler.

Selv om det ikke var uventet, kom det underveis i prosjektet flere hindringer. Prosjektet var en konseptutprøving og ikke nødvendigvis ment til å bli perfekt. Med dette som utgangspunkt var ikke prosjektet planlagt i detalj og dermed var også en stor del av oppgaven til studentgruppen å håndtere disse uventede hendelsene så godt det lar seg gjøre underveis. Studentene traff etter hvert utfordringer som gjorde at de måtte bytte retning da prosjektet ikke lengre var gjennomførbart i henhold til den originale spesifikasjonen. Til tross for dette fortsatte studentene arbeidet og har lagd ett produkt.

Jeg må si meg imponert over hvordan studentgruppen har gått frem gjennom prosjektet og håndtert de ulike situasjonene. Ukentlige statusmøter arrangert av studentgruppen, tydelig skriftlig kommunikasjon i forveien og god oppfølging har gjort at et konseptprosjekt mer trygt. Jeg som oppdragsgiver kan se hvordan gruppen har tilegnet seg vesentlig kunnskap gjennom prosjektet samt en god evne til å jobbe sammen.

Med vennlig hilsen

Kevin Benjamin Zeppo Adriaansen

Vedlegg 2: Egenvurdering

Gruppen:
Vi har møtt på utfordringer og samtlige medlemmer har vært like motiverte på å løse og tilpasse seg til dem. Gruppen har arbeidet sammen og vært stort sett den samme i flere semestre. Dette har gjort oss godt kjent, bygget en god kultur i gruppen og lært oss å kjenne hverandres styrker og svakheter. Ingen i gruppen er redde for å dele personlige meninger og tanker, og vi fungerer like bra sammen sosialt på fritiden. Kollektivt har gruppen fungert strålende gjennom vårsemesteret.

Christian Thien Truong Luong Nguyen
løpet av dette prosjektet har jeg blitt utpekt til gruppeleder og scrum-master. Når det gjelder ansvar som gruppeleder har jeg for det meste bare hatt ansvar for å booke rom/møter og ha dialog med oppdragsgiver eller andre interessenter. Som scrum-master hadde jeg ansvar for ClickUp/Jira og overvåkning av verktøyet, samt organisering av scrum-aktiviteter som planning poker og daily scrums. I tillegg til dette har jeg også bidratt til rapportskriving og da design også var i fokus deltok jeg i å utvikle wireframes.

Edi Grajcevci
I dette prosjektet har jeg hovedsakelig jobbet med backend utvikling av AI. Fokuset mitt har vært på implementeringen av både open source og closed source løsninger i prosjektet. Jobbet mye med LlamaIndex og anvendingen av rammeverket RAG med ulike Llama og GPT modeller, samt fine tuning i starten av prosjekt løpet.

Huy Trong Vu
Gjennom dette prosjektet har hovedfokuset mitt vært på utviklingen av back-end-koden for AI-en. Her har fokuset mitt vært på Open Source AI og lokale språkmodeller. Jeg har jobbet med å finne og teste ulike språkmodeller på Hugging Face, som Llama 2, Yi og Phi-2. Samt har jeg jobbet med å design på Figma, og da brukertestene.

Trym Falkum
Mitt bidrag til bachelorprosjektet har vært gjennom alt fra rapportskriving til system-prompting. Personlig liker jeg å være delaktig i de fleste områdene av prosjektet. Det jeg har hatt hovedansvar for selve rapporten, og fungert som en sekretær ved dokumentasjonen av notater generelt, notater for møter og andre viktige hendelser, samt hatt ansvar for store deler av kommunikasjonen med interessentene. I tillegg har jeg hatt ansvar, sammen med Huy, om forberedelser, utføring og analyse av intervjuet og brukertestene.

Isak Meen

Gjennom arbeidet med bacheloroppgaven har jeg bidratt på flere områder. I designfasen har jeg utviklet wireframes med Huy på Figma. Videre har jeg skrevet og redigert flere deler av rapporten, og sørget for at den følger akademiske standarder. Jeg har også notert viktige punkter i statusmøter, noe som har sikret god dokumentasjon og struktur i prosjektet. I tillegg har jeg hatt en aktiv rolle i idemyldring, hvor jeg har foreslått kreative og gjennomførbare løsninger, samt rensing og formatering av kunnskapsbase.

Vedlegg 3: Liste over verktøy og metoder

Vedlegg 4: Første svar fra RAG

{'query': 'Hvordan kan jeg gjøre en fagperson synlig i nedtrekkslister på flere fakultet?',

'result': ' You can make a fagperson visible in withdrawal lists on multiple faculties by using the "Und." semester option in the faculty profile settings. This will move the fagperson from the "Und." semester to the current semester, making them visible in withdrawal lists for all faculties.'}

Vedlegg 5: Tentativ tidsplan

TENTATIV TIDSPLAN

Felles for alle sprinter:

Testing av ulike komponenter og artefakter, spesielt AI og brukerveiledningene
Dokumentering av arbeid
Rapportskriving
Daily scrums / Stand-up møter
Ukentlig statusmøte med oppdragsgiver
Minst ett møte med veileder annenhver uke
Noe form for teambuilding

*Markert i gult = nedprioritert

Sprint 1: (23. januar - 16. februar)

Idémyldringsfasen

- Grunnlegge forståelse av prosjektet og domenene
- Brainstorming
- Backlog

Planleggingsfasen

- Kartlegge behov: MoSCoW
- Brukerreise

Kort begynnelse av designfase, fortsettes neste sprint

- Alle gruppemedlemmer lager et design/sketch.

Testing og kalibrering av AI

- "Leke" litt med AI, gjøre det klart til implementering. Testing av svar fra AI, dette skjer iterativt i alle sprinter.

Frister sprint 1:
-Styringsgruppemøte nr. 1 med oppdragsgiver uke 7+-
-Statusrapportering 14. februar

- Alle grupper presenterer status

Sprint 2: (19. februar - 1. mars)

Første versjon av design, vises frem i statusmøte for godkjenning/forbedringer

- Wireframe
- Mockup

Videre intensiv arbeid med AI

Frister sprint 2:
-

Sprint 3: (4. mars - 22. mars)

Fullføre første "prototype"

Frister sprint 3:
-Midtsemesterevaluering Uke 10
-Statusrapportering 20. mars

- Alle grupper presenterer status

Sprint 4: (25. mars - 29. mars)

"Catch-up"- allokerer tid til feilberegnet oppgaver.

Brukertesting

- Analyser resultatene, forbedringer

Påskeferie - dersom ting ligger på god plass

Frister sprint 4:
- Styringsgruppemøte nr. 2 med oppdragsgiver uke 13 eller 15+-

Sprint 5: (1. april - 19. april)

Iterasjon
Videreutvikling av prototypen

Frister sprint 5:

Sprint 6: (22. april - 26. april)

Siste brukertesting

- Validering av prosjektet

Refleksjon av prosjektet

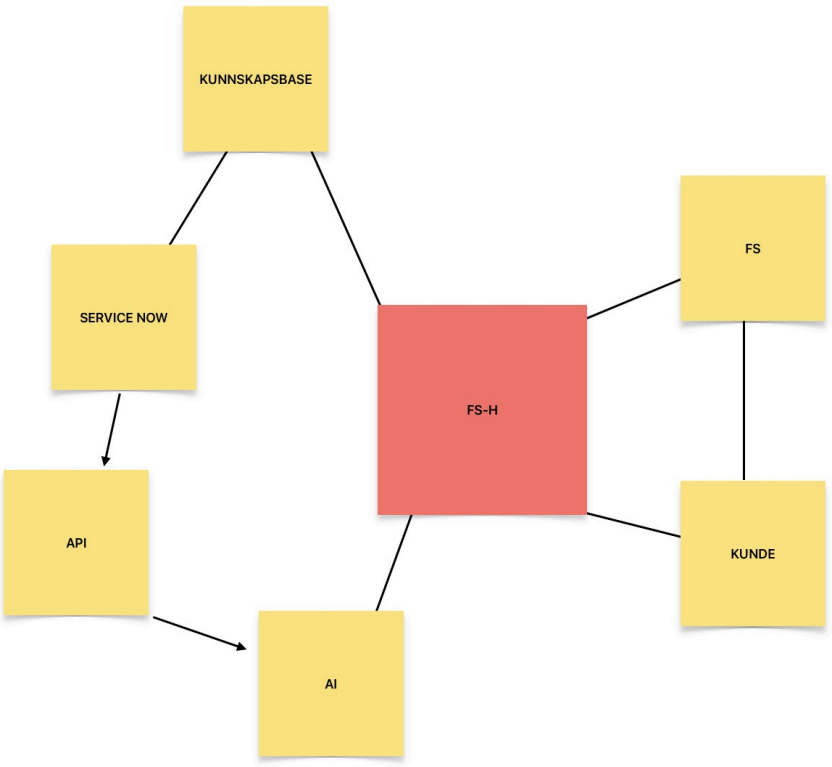
- Hva funket/funket ikke? Hvorfor? Hva bør gjøres videre?

Resten av tiden vil fokuseres på å fullføre rapporten (29. april - 16. mai)

Frister sprint 6:

Frister utenfor sprintene:
-Sluttrapport Bachelor IS-304 til 16. mai
-Registrer prosjektet på Kompetansetorget 16. mai
-Expo 22. mai
-Muntlig eksamen 3,4,6 og 7 juni

Vedlegg 6: Tankemodell versjon 1



Vedlegg 7: MoSCoW-metoden

Som en …	**Ønsker jeg å…**	**Slik at …**	MoSCoW (prioritering)
Bruker	Spørre om brukerstøtte	Jeg kan å løse utfordringene jeg har med FS	Must have
AI	Gi brukerstøtte	Jeg kan løse utfordringene til brukeren uten ventetid på saksbehandler	Must have
Bruker	Sende oppfølgingsspørsmål	Jeg kan gi mer kontekst for bedre svar eller spørre om problemer som dukker opp	Should have
Saksbehandler	Motta sak fra bruker	Jeg kan gi de brukerstøtten som AI ikke kan løse	Could have
Bruker	Få forslag på oppsett til hvordan jeg skal spørre om brukerstøtte	Jeg kan få bedre brukerstøtte uten å sende oppfølgingsspørsmål med mer kontekst	Won’t have
Saksbehandler	Få daglig rapport av historisk data	Jeg kan arkivere data med hensyn til universitetets retningslinjer	Should have
AI	Kritisere dårlige brukerveiledninger	Jeg kan gi tilbakemelding om veiledning er for dårlig beskrevet eller har for lite kontekst og trenger forbedring.	Should have (vanskelig)
Bruker	Velge kategori på brukerstøtte	Jeg kan få mer spesifikk brukerstøtte	Won’t have
Bruker	Ha oversikt over mine tidligere saker og samtaler med AI	Slik at jeg kan få oversikt over eller følge opp på tidligere saker og samtaler med AI	Should have
Bruker	Jeg kan sende saken videre til saksbehandler	Jeg kan få brukerstøtte til det AI ikke kan løse.	Should have

Vedlegg 8: Risikoanalyse

Risiko	Konsekvens (1 til 5)	Sannsynlig-het (1 til 5)	Risiko	Tiltak for at ikke skjer	Tiltak for å redusere skade
API-nøkkel blir fanget opp	4	2	8	Skjule API-nøkkelen og rotere nøkkel periodisk	Slette API-nøkkelen og erstatte den med en ny
AI-assistent gir dårlig støtte	3	2	6	Gi LLM kvalitetsdata og gjennomfør tilstrekkelig testing	Mulighet til å følge opp på spørsmål, eller gå videre til en saksbehandler
Manglende kunnskap for å fullføre prosjektet	4	5	20	Kontinuerlig læring innenfor domenet., Spørre om hjelp fra veileder. Hjelpe hverandre i gruppen	Tilby tilstrekkelig med dokumentasjon på hva som er gjort så prosjektet kan videreutvikles
Manglende dialog med oppdragsgiver	4	1	4	Holde god kontakt med oppdragsgiver og fortsette med ukentlige møter	Sette opp møter og gjenopprette dialog
Ikke bli ferdig med prosjektet i tide	4	2	8	Fokuser på de viktigste funksjonalitetene, legg til flere funksjoner dersom det er tid. (MoSCoW)	Se “Manglende kunnskap for å fullføre prosjektet”
Sosial loffing	4	1	4	Jobbe sammen i grupperom, Daily Scrum og gruppekontrakt	Ta opp problemet med personen og gi de en mulighet til, kanskje de trenger noe annet å jobbe med isteden. Dersom dette ikke gir seg, anvend til gruppekontrakten og konsekvensene.
Dårlig kodekvalitet	3	2	6	Ha felles kodestandard, gjennomfør code	

				reviews	
Lite motivasjon i gruppen	5	1	5	Ta gode pauser, ha klare og mulige mål, ha team building i blant.	Si ifra tidlig til gruppen dersom motivasjonen er lav, hopp over til en mer utfordrene eller interessant oppgave.
Noen i gruppen blir syk	1	5	5		Mulighet til å jobbe hjemmefra.
Noen i gruppen blir alvorlig syk	4	2	8		Gi oppgavene til personen som er syk til resten av gruppen.
Uenigheter som hindrer fremgang	4	1	4	Gruppen har en flat struktur og om uenigheter i gruppen oppstår er vi fem medlemmer, noe som betyr at flertallet alltid vil havne på en avgjørelse i demokratiske avstemninger	Ta opp uenigheter i gruppen tidlig slik at det kan diskuteres og løses.
Kommer for sent til møtet	3	2	6	Ha god oversikt over møter, sett opp på kalender.	Si gjerne ifra om man kommer forsinket til møtet. Dette er greit om det er i få instanser, men dersom dette skjer ofte må det bli tatt en samtale om. Hvis dette ikke løser seg, henvis til gruppekontrakt og konsekvensene.

Vedlegg 9: Utsnitt av dagbok til sprint 2

Dagbok:

SPRINT 2

19. februar
Hele gruppen:
6 timer sprint 2 planning. Planla hva som skal gjøres i Sprint 2. Planla litt arbeidsfordeling slik at gruppen vet hvem som skal gjøre hva, og hvem som skal jobbe sammen denne sprinten. Mye av tiden gikk til å sette opp ClickUp, men måtte gå over til ny plattform - Jira, fordi det plutselig kom en betalingsmur.

20. februar
Huy:
4 timer wireframe på figma, hjelp til når vi skal utvikle programmet. får også reflektert hva som er nyttig å ha med og hva som kan droppes / gjøres enklere

Christian:
3 timer til å fikse backlog på Jira. La inn flere sprint items og satt opp story point estimates for å få bedre oversikt over hvordan vi skal jobbe med sprinten og kommende sprinter. Måtte flytte over fra Click-Up siden det kostet penger å legge til felt til items/taskene og derfor krevde det mer tid i dag for å fikse dette. Jira er det vi kommer til å bruke fremover i prosjektet, det virker lovende ut.
1 time på wireframe på figma, hjalp til med en frame som skulle bli brukt til å fremvise hvordan det kunne sett å ha et oppfølgingssystem på saksgrensesnittet.

Trym:
4 timer rapportskriving. Satt opp flere kapitler i rapporten for struktur i rapport, begynt på kapittelet som foreløpig heter Prosjektgjennomføring under delkapitlet Prosjektstart (for å komme videre på rapporten).

Isak:
4 timer wireframe og backlog.
Tegnet en interface for dialog med fs-hjelp. Dette er fordi vi trenger å ferdigstille for å fortsette med prototype. La til tasks på nye backlog systemet vårt Jira, konkrete oppgaver for laging av frontend views med story points for at Jira sitt system skal kunne lage et fungerende Burndown chart

Edi:
4 timer med research på AI og sett mer på funksjonalitet rundt AI. Satt opp miljø for å starte på selve applikasjonen. Også startet å se på andre detaljer som docker og hvordan vi kan teste AI lokalt og lese på mer dokumentasjon av AI fra Microsoft. Funnet ut av caching og at lagring av samtaler kan gjøre rett fra OpenAI sin løsning, men kan hende det kreves justeringer om brukerveiledninger fra AI blir for lange for "token count". Utforsket huggingface grunnet at gpt 3.5 turbo fine-tunings modellen kan føre til store økonomiske kostnader for universitetet, men det var veldig mye informasjon å ta inn så dette krever mer research. For nå fortsetter vi med gpt 3.5 turbo fine-tunings modellen.

Vedlegg 10: Intervjuguide - semistrukturert intervju med brukertesting

Torsdag, 7. mars, 2024

Dybdeintervju med brukertesting

Før produktgjennomgang:

- Hva er rollen din i FS?

  (Saksbehandler eller bruker av FS Hjelp)
- Hvor ofte bruker du FS?
- Er du godt kjent med systemet?
- Er det ofte du sliter med å bruke FS?

Hvis ja:

- Klarer du å spesifisere hvor eller hvorfor?

  (F.eks. at det er vanskelig å navigere eller forstå)
- Har du fått veiledning fra FS-Hjelp/brukerstøtte tidligere?

Hvis ja:

- Hvor lang tid tok det før du fikk hjelp med saken?
- Var det for lenge?
- Man ønsker gjerne å få svar så fort som mulig, men hvor lenge kan du vente på svar før det blir for sent?
- Hvor ofte trenger du hjelp fra FS-Hjelp/brukerstøtte?
- Hadde du brukt en web-applikasjon som bruker kunstig intelligens til å gi deg automatisk brukerstøtte om det trengs?

Hvis ja:

- Hva slags funksjonalitet vil du ha i en slik applikasjon?