Høgskulen på Vestlandet

BACHELOROPPGAVE

Webapplikasjon for administrasjon av bedriftsressurser

Web application for managing corporate resources

Elisabeth Marie Hovden

Anders Benjamin Grinde

Endre Tuft Botnen

Data / IT

Institutt for data- og realfag

Fakultet for ingeniør- og naturvitskap

Innleveringsdato: 03.06.2019

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

Høgskulen på Vestlandet

Fakultet for ingeniør- og naturvitskap

Institutt for data- og realfag

TITTELSIDE FOR HOVEDPROSJEKT

Rapportens tittel: Webapplikasjon for administrasjon av bedriftsressurser	Dato: 03.06.2019
Forfatter(e): Elisabeth Marie Hovden Anders Benjamin Grinde Endre Tuft Botnen	Antall sider u/vedlegg: 39 Antall sider vedlegg: 10
Studieretning: Data / IT	Antall disketter/CD-er:
Kontaktperson ved studieretning: Bjarte Kileng	Gradering: Ingen
Merknader:	

Oppdragsgiver: Scout Gaming Group	Oppdragsgivers referanse:
Oppdragsgivers kontaktperson: Stian Lysvold Haugse	Telefon: 90 66 99 18

Sammendrag:

Denne rapporten beskriver prosessen av å utvikle en webapplikasjon for bedriften Scout Gaming Group, samt bakgrunnen for oppgaven. Resultatet av prosjektet, en webapplikasjon med alle nødvendige funksjoner for administrasjon, skal brukes internt hos bedriften til administrasjon av bedriftsressurser.

Stikkord:

Webapplikasjon	Python	PostgreSQL

Høgskulen på Vestlandet, Fakultet for ingeniør- og natuvitskap
Postadresse: Postboks 7030, 5020 BERGEN Besøksadresse: Inndalsveien 28, Bergen
Tlf. 55 58 75 00 Fax 55 58 77 90 E-post: post@hvl.no Hjemmeside: http://www.hvl.no

Forord

Denne bachelorrapporten er skrevet av Elisabeth Marie Hovden, Anders Benjamin Grinde og Endre Tuft Botnen. Rapporten dokumenterer arbeidet bak og bakgrunnen for vår bacheloroppgave: *Webapplikasjon for administrasjon av bedriftsressurser.*

Først og fremst ønsker vi å rette en takk til Scout Gaming Group for å ha gitt oss denne oppgaven, da spesielt Stian Lysvold Haugse som har vært vår eksterne veileder og kommet med gode tilbakemeldinger underveis. Takk for et godt samarbeid og for gjestfriheten i bedriften.

Takk til Helene Dunlop og Mikal Meltvik for å ha hjulpet oss med diverse tekniske problemer underveis og til alle de ansatte i bedriften for et godt arbeidsmiljø gjennom disse månedene.

Vi ønsker også å takke Bjarte Kileng, vår interne veileder på HVL, for veiledning gjennom prosjektperioden og verdifulle tilbakemeldinger.

Innholdsfortegnelse

1 INTRODUKSJON

Denne delen er en introduksjon til prosjektet. Den vil gi et godt overblikk over mål og motivasjoner til prosjektet, samt beskrive innhold, begrensinger og ressurser.

1.1 Mål og motivasjon

Målet med denne oppgaven er å lage et system for Scout Gaming Group (SGG) som holder oversikt over alt utstyr som blir kjøpt inn i bedriften. Systemet vil være med på å forbedre hverdagen til de ansatte som jobber i bedriften, gi en god struktur og en oversikt som er enkel og brukervennlig. Oppdragsgiver, Stian Lysvold Haugse, hadde en tanke om hvordan han ville ha det og hvilke funksjonaliteter som skulle være mulig i dette systemet.

Det ferdige produktet skal være en webapplikasjon som skal kunne integreres med deres intranettsystemer. Det er derfor blitt valgt å skrive selve prosjektet i Python ved hjelp av rammeverkene Flask og SQLAlchemy. Databasen vil bli skrevet i PostgreSQL for at SGG skal kunne integrere det lett med sine egne servere.

Løsningen skal ha mulighet for å legge til nye ansatte, beskrive hvilken avdeling de jobber på og la alle ansatte være registrert med ansatt ID. Det skal også være mulighet for å se en oversikt over alle ansatte i bedriften.

Hoveddelen av applikasjonen skal være en oversikt over alt utstyret som SGG har kjøpt inn. Oversikten vil kunne sorteres og det skal være mulig å registrere hvem som har utstyret akkurat nå. Man skal kunne se en oversikt over hvilket utstyr en person har akkurat nå og hvem som har hatt utstyret tidligere.

Motivasjonen vår ligger i at den løsningen som SGG har per dags dato ikke er god nok, og med mye data registrert blir den veldig uoversiktlig. Stian hadde et stort ønske om å få endret dette, slik at han kan få brukt mer tid på andre ting i arbeidsdagen.

1.2 Kontekst og relevans

Måten SGG håndterer utstyrslisten til hver ansatt i dag, er ved hjelp av et Excel-ark med flere faner. Dette Excel-arket blir brukt som en database og inneholder en oversikt over alt elektronisk utstyr i bedriften og en liste over hvem som er tildelt hva. Listen blir manuelt oppdatert av Stian, noe som er ressurskrevende. Dette skyldes et uoversiktlig oppsett, som fører til at det er tidkrevende å legge til nye data og legge til nødvendige referanser. Løsningen de har i dag fører til at det er vanskelig å ha en god oversikt, og vil ikke være skalerbar sammen med bedriftens vekst.

Siden dagens løsning er ressurs- og tidkrevende ønsker Stian å komme seg vekk fra dette systemet, og heller bruke et nytt system som er mer skalerbart og oversiktlig. En webapplikasjon kan løse problemet og gjøre det enkelt å legge til nye data.

1.3 Begrensinger

Det er et par begrensninger som må tas i betraktning ved utvikling av en slik webapplikasjon.

Tilgjengelige arbeidstimer

Det er ikke alltid hele gruppen har mulighet til å stille, og det er ingen som kommer til å være i SGG sine lokaler hver dag i uken. Vi må derfor være flinke til å kommunisere og med å være effektive når det settes av tid til å jobbe med prosjektet. Vi er også nødt til å velge hvilke områder av applikasjonen vi ønsker å fokusere på. Her må det gjøres en vurdering på hva som er viktig for at applikasjonen skal være funksjonell, og hva som bare hadde vært greit å ha som tilleggsfunksjoner.

Forkunnskaper i gruppen

Prosjektgruppen har forskjellige forkunnskaper, og ingen i gruppen har jobbet med Flask i Python før. Det er derfor satt av tid til å sette seg inn i dette før prosjektet begynner.

1.4 Ressurser

Bedriften stiller med flere ressurser som kan benyttes gjennom prosjektperioden. Noen av disse ressursene er lokaler, utlånt utstyr, mat og drikke. I prosjektet vil det bli holdt et tett samarbeid med bedriftens ansatte. Disse ansatte vil bistå med ressurser i form av kunnskap. Helene Dunlop vil være vår tekniske veileder. Hun har jobbet som utvikler i SGG i snart to år, og er godt kjent med de fleste verktøy og programmeringsspråk som bedriften benytter seg av.

Lokalene er lagt opp for et godt samarbeid mellom de ansatte og oss som studenter. Vi vil disponere egne PCer gjennom prosjektet, men har også tilbud om lån av stasjonære PCer hos bedriften.

Vi har også tilgang på HVL, Høgskulen på Vestlandet, sine lokaler, hvor vi kan få hjelp av veilederen vår Bjarte.

1.5 Organisering av rapporten

Rapporten er lagt opp i 10 kapitler som gir en beskrivelse av løsningen som har blitt presentert for SGG.

- Kapittel 1 gir en oversikt over og introduksjon av oppgaven og løsningen som er presentert. Det gir også en kort oversikt over ressursene som er tilgjengelige for gruppen.
- Kapittel 2 er en beskrivelse av prosjektet i detaljer, forslag og krav som Stian har til prosjektet.
- Kapittel 3 beskriver hvordan det har blitt valgt å utforme prosjektoppgaven, hvilke rammeverk og verktøy som er tatt i bruk, og hvilke programmeringsspråk prosjektet skal skrives i.
- Kapittel 4 gir en detaljert beskrivelse av applikasjonen og vil gi en god forståelse av hvordan applikasjonen ser ut og hvordan den fungerer i praksis.
- Kapittel 5 viser hvilke metoder som har blitt brukt for å evaluere dette prosjektet og hvordan det har blitt evaluert om det ferdige produktet holder mål for Stian og SGG. Kapittelet viser også resultatene fra evalueringene.
- Kapittel 6 inneholder en refleksjon rundt og informasjon om resultatet av prosjektet. Kapittelet vil også gi mer informasjon om evalueringsresultatene.
- Kapittel 7 er en diskusjon om hvordan prosjektet har gått og hva som kunne vært gjort annerledes.
- Kapittel 8 gir en kort konklusjon av prosjektet.
- Kapittel 9 er en liste over kilder og referanser som har blitt brukt i denne rapporten.
- Kapittel 10 inneholder en liste over risikoer og diagrammer for prosjektet, samt alle andre relevante vedlegg.

2 PROSJEKTBESKRIVELSE

2.1 Bakgrunn

2.1.1 Oppdragsgiver

Oppdragsgiveren for oppgaven er Scout Gaming Group. SGG er et konsern som ble startet av to pokerspillere fra Bergen i 2013. De lager og drifter en spillplattform som brukes til «Fantasy Sports» og «pool betting»-spill.

Kundene deres er andre spillselskap som ønsker å tilby Fantasy Sports til sine spillere. Eksempler på dette er Betsson og Norsk Tipping. De leverer også spill, som VM-Manager, til medieselskaper som TV2 og NRK.

SGG har per i dag ca. 80 ansatte fordelt på kontorer i Bergen, Stockholm, Ukraina og Malta.

2.1.2 Tidligere arbeid

SGG har en nåværende løsning hvor de bruker et Excel-skjema for å føre opp en oversikt over utstyret de har gått til innkjøp av. Kun i 2018 kjøpte de datautstyr for ca. 700,000kr så dette er en post det blir stadig viktigere for de å ha god oversikt over.

	A	B	C	D	E	F	G	H	I	J
1	ID	receipt	item	bought	price	model	description	employee	note	Sold/died
2	201459-1	59	1	23.01.2014	kr 3 196		P8 DT Pen-G3220/12G/1T/GT62			
3	201459-2	59	2	23.01.2014	kr 3 196		P8 DT Pen-G3220/12G/1T/GT62			
4	2014247	247		23.02.2014	kr 6 517	2013	Asus N56JR 15,6" i7/8GB/750GB HDD/SSD	Gone	ødelagt skjem	
5	2014473	473		11.11.2014	kr 5 192	2013	Komplett Office i7-4770S/8GB/120GB	Andreas Sundal		
6	2014X	X		01.12.2014	?	2014	Macbook Pro i5 8GB	Loken Makwana	Kjøpt i desember på ebay iflg. Loker	
7	2015160-1	160	1	16.04.2015	kr 5 432	2014	Mac mini MGEN2D		Lviv	
8	2015160-2	160	2	16.04.2015	kr 5 432	2014	Mac mini MGEN2D		Lviv	
9	2015503	503		16.04.2015	kr 5 432	2014	Mac mini MGEN2DH/A		Lviv	
10	2015222	222		23.04.2015	kr 6 399	2014	Asus UX305FA-FC004H 13.3" laptop		Andreas?	
11	2015461	461		29.06.2015	kr 11 992	2014	Macbook Pro 15" MGXA2HA	Torbjørn Angeltveit	knust skjerm, brukes som stasjonær	
12	2015449-1	449	1	01.09.2015	kr 6 072	2014	Iphone 6 64GB Space Gray			
13	2015449-2	449	2	01.09.2015	kr 3 112	2013	Iphone 5C 8GB Blue			
14	2015868-2	868	2	01.09.2015	kr 4 444		LG 55"LED/FHD/50Hz/T2CS2	Office - Bergen	Utviklerrommet	
15	2015868-3	868	3	01.09.2015	kr 4 444		LG 55"LED/FHD/50Hz/T2CS2	Office - Bergen	Utviklerrommet	
16	2015868-1	868	1	01.09.2015	kr 4 444		LG 55"LED/FHD/50Hz/T2CS2	Storage - Bergen		
17	2015X	X		16.12.2015	kr 7 236		Lenovo Y50-70 15.6"	Espen Rød	kjøpt i 2015, refusjon i 2018	
18	2016394-1	394	1	10.06.2016	kr 5 592	2016	Samsung Galaxy S7 32GB White			
19	2016394-2	394	2	10.06.2016	kr 3 992	2014	Iphone 6 White 16			
20	2016394-3	394	3	10.06.2016	kr 3 992	2014	Ipad Air 2 64GB Wifi Grey			
21	2016AP121	AP121		21.06.2016	kr 12 791	2015	Macbook Pro 13,3" MF840HA	Tony A. Sæle		
22	2016BA234-1	BA234	1	11.07.2016	kr 6 152	2014	Mac mini MGEN2D		Lviv	

Figur 2.1: Utdrag fra dagens løsning hos SGG

Den nåværende løsningen er ikke enkel å bruke og gjør at bedriften lett mister oversikten dersom noe ikke blir registrert som det skal. Alt må legges inn manuelt - det innebærer at de manuelt må legge til en ny ansatt, manuelt må registrere at en ansatt har et utstyr og deretter

manuelt registrere selve utstyret med tilhørende data i et siste regneark. Dette er en veldig tungvint måte å gjøre det på og det krever mye unødvendig tid og krefter, samt øker sjansen for å gjøre menneskelige feil i systemet.

Det er mye repetering av samme data som burde være automatisert og det er dermed lett å bruke mye tid på finne ut av det som trengs. Dagens løsning er rotete og lite oversiktlig, noe dette bachelorprosjektet vil være med på å endre.

2.1.3 Kravspesifikasjon

Webgrensesnittet og databasen må kunne gi følgende funksjoner og rapporter:

Registrere nytt utstyr og eventuelt slette utstyr som er registrert feil

Applikasjonen skal ha støtte for å legge til utstyr i en database og gi en oversikt over disse dataene. På oversikten vil det være mulighet for å trykke på et utstyr og få en oversikt over det, deretter vil det være mulig å slette det hvis det er registrert feil. Det burde også være mulighet for å kunne redigere data som allerede er registrert.

Laste opp kvittering eller lenke til kvittering i Google Drive

I første iterasjon ble det valgt å lage en tabell i databasen som inneholder informasjon om kvitteringene. I denne tabellen er det mulig å legge til en lenke til Google Drive. Det er ikke mulig å laste opp selve kvitteringen, men et felt med en lenke til Google Drive var en god nok løsning.

Registrere hvem som har fått hvilket utstyr

Det å registrere hvem som har fått hvilket utstyr er en del av hovedoppgaven til applikasjonen og følgelig noe som settes høyt på prioriteringslisten. Denne funksjonaliteten må være på plass for at det skal være mulig å si hvem som har gammelt utstyr, om noen som har sluttet fortsatt har noe utstyr og hvem som har hatt utstyret tidligere.

Få en oversikt over hvilket utstyr en ansatt har

Applikasjonen vil ha mulighet for å gi en oversikt over alt utstyret som en ansatt har tilgjengelig akkurat nå.

Få en oversikt over hvilket utstyr som er ledig og hvor det befinner seg

Det må være mulighet for å vise oversikten over alt utstyret som ikke er registrert på en person. Det oversikten vil si noe om er kun hvilken avdeling det befinner seg på, og kommer ikke til å bli gjort mer spesifikt enn det.

Få oversikt over hvem som har hatt utstyr siden det ble kjøpt

Utstyret vil ha en transaksjonshistorikk. Denne historikken vil kunne vises til brukeren, slik at det er mulig å se hele oversikten over hvem som har hatt utstyret tidligere.

Få varsel om hvilket utstyr som snart er klart for utskiftning

Det vil bli snakket med Stian mer spesifikt om når noe skal defineres som klart for utskiftning, men dette er noe som applikasjonen skal ha støtte for. Det er viktig at personene som jobber i et IT-selskap som SGG har oppdatert utstyr for å kunne gjøre jobben sin best mulig.

Få varsel om utstyr som mangler kvittering, bilag eller lokasjon

I første omgang kommer det til å være slik at et utstyr mangler lokasjon når det ikke har blitt tildelt noen personer, og plasseres under en "ikke registrert"-liste. Det ønskes også å legge til støtte for advarsler dersom noe av informasjon mangler på et registrert utstyr.

Utstyr må kunne registreres som ledig, selv om det er tildelt en ansatt

Selv om et utstyr er tildelt en ansatt burde det være mulig å registrere det som ledig. Dette er noe som ligger litt bak på prioriteringslisten, og det vil i de første iterasjonene ikke være støtte for dette. Hvis det er tid, så ønsker gruppen at det skal være støtte for dette i applikasjonen.

2.1.4 Opprinnelig forslag

Det ble lagt frem et ønske om å lage en oversikt over alle innkjøp av utstyr som ble gjort i SGG og hvem som har eierskap til det i dag. Forslaget som gruppen la frem var å lage en webapplikasjon som skulle kunne integreres med systemene som allerede eksisterer i dag. Applikasjonen skal ha en innloggingsside slik at ikke hvem som helst skal kunne logge seg inn og legge til eller fjerne utstyr. Denne skal i hovedsak bare Stian ha tilgang til, men muligheten for å legge til flere brukere skal være til stede.

Figur 2.2: Stian sitt opprinnelige forslag, hentet fra PowerPointen til bedriften

Etter innlogging vil brukeren komme til en oversiktsside. Her vil det være ulike advarsler definert, som vist i figuren over. Herfra skal det være mulig å trykke seg videre til sider med oversikt over utstyr, ansatte, transaksjoner, kvitteringer og departementer.

I oversiktene til de forskjellige delene skal det være mulig å sortere listen, det skal også være mulig å legge til et nytt utstyr og registrere hvem som har det. Øverst vil det være en meny som vil kunne ta brukeren til de forskjellige andre delene av applikasjonene. Utstyret skal kunne lenkes til en kvittering slik at det er mulig å se selve kvitteringen for kjøpet. Ved klikk på et utstyr skal det være mulig å se transaksjonshistorikk, det vil si hvem som har hatt utstyret før, hvem som har utstyret akkurat nå, og om det eventuelt er blitt solgt eller skrotet. Under oversiktslisten vil det være lister over utstyr som ikke er tildelt noen personer og en liste over utstyr som ikke er registrert som returnert fra en person som har sluttet.

Det skal være en oversikt over de ansatte og hvilke utstyr de har akkurat nå. Likt som i utstyrsoversikten skal det være mulighet for å trykke på en ansatt for å se hvilket utstyr de er tildelt. Det vil på samme måte som i utstyrsoversikten her være mulighet til å legge til en ny ansatt i databasen. Nederst på siden skal det være en liste over personer som ikke har utstyr.

2.2 Litteratur bakgrunn

Litteraturen som har blitt benyttet som kilder i denne oppgaven er hentet fra diverse nettsider og elektroniske leksikon. Informasjon om rammeverk, programmeringsspråk og verktøy har blitt hentet fra disse utgivernes offisielle nettsider. Annen informasjon, som blant annet informasjon om utviklingsmetoder, har blitt hentet fra nettsiden til anerkjente bedrifter. Vi har vært kritiske til hvilken informasjon vi har benyttet oss av og gjerne sammenlignet informasjon fra ulike distributører for å sikre at feilinformasjon ikke ble gjengitt her i denne bacheloroppgaven.

3 PROSJEKTDESIGN

3.1 Mulige fremgangsmåter

Det finnes flere tilnærminger til problemet som kan vurderes for å løse oppgaven. De ulike tilnærmingene har både fordeler og ulemper med seg. For å kunne lage ønsket webapplikasjon spesifisert av oppdragsgiver har vi vurdert to ulike fremgangsmåter.

3.1.1 Lage et forbedret Excel-ark

Et nytt Excel-ark, selv i forbedret versjon, var i grunnen aldri et alternativ da oppdragsgiver spesifikt ønsket en webapplikasjon. Om det derimot skulle vise seg at det ikke var mulig å løse det på ønskelig format, hadde en slik forbedret variant vært noe vi kunne vurdert.

Fordelen med en slik løsning er at det hadde vært lite tidkrevende å lage og at alle gruppemedlemmene på prosjektgruppen hadde de nødvendige forkunnskapene fra før av.

En annen fordel er at oppdragsgiver har brukt Excel i flere år og har god kjennskap til programmet. Det ville dermed ikke vært nødvendig med en innføring, og det ville enkelt vært mulig for han å legge til ny funksjonalitet med Excel sine forhåndsdefinerte funksjoner. Samtidig ville det heller ikke vært nødvendig å bruke tid på å flytte eksisterende data over til det nye systemet.

Ulempen er at det ikke ville vært like store muligheter for å legge inn ekstra funksjonalitet, utover det Excel selv kan tilby, og å bygge noe som var like skalerbart. Det ville heller ikke oppfylt de fleste kravspesifikasjonene som oppdragsgiver kom med og kunne dermed ikke gitt et godkjent sluttresultat.

3.1.2 Lage en webapplikasjon

En webapplikasjon åpner for gode muligheter for implementering av det man måtte ønske av funksjonalitet. Fordelen med en slik løsning er at man står relativt fritt til å lage det slik man ønsker. Det ligger også mye kode tilgjengelig ute på nettet som kan gjenbrukes, da spesielt i forbindelse med designutformingen.

Ulempen er den ekstra tiden det krever å bygge en webapplikasjon. Sammen med større muligheter kommer også en større sannsynlighet for feil i programmet, funksjoner som ikke fungerer slik de skal, og alt i alt mer arbeid for å få ting til å virke.

Det vil også være tidkrevende å flytte over allerede eksisterende data til det nye systemet. Den gamle løsningen har data datert helt tilbake til starten av 2014, noe som vil si at det er over fem år med data som skal overføres.

Samtidig vil mengden med data som lagres bare øke, og det er derfor bedre å ta tak i problemet nå enn å vente til den nåværende løsningen absolutt ikke er brukbar lenger.

Vi har også vurdert ulike programmeringsspråk/rammeverk for den siste fremgangsmåten. Den fullstendige listen ligger under punkt 3.3 hvor vi også har inkludert verktøyene vi valgte å benytte oss av.

3.1.2.1 Python Flask

Python Flask er et Python-mikrorammeverk som blir brukt for å lage webapplikasjoner (Flask, u.å.). Flask-rammeverket trenger ikke ekstra verktøy eller biblioteker for å kunne bli brukt, kun standardbiblioteket. Den største fordelen med Flask er at det er enkelt å lære seg og at man kan raskt sette opp en fungerende webapplikasjon. Koden i Flask er enkel å vedlikeholde siden det er få integreringer som må oppdateres.

Forkunnskaper

Prosjektgruppen har ingen forkunnskaper i rammeverket Flask, men alle på gruppen har jobbet med Python tidligere.

3.1.2.2 Node.js

Node.js er et JavaScript-rammeverk. Node.js er et godt rammeverk for applikasjoner med stor skalering. Det finnes et bredt spekter av APIer som er tilgjengelig for dette rammeverket. Webapplikasjoner som bruker dette rammeverket har som regel høy hastighet, og oppgaver kan bli utført asynkront. Det er perfekt for applikasjoner som kjører oppgaver i sanntid og med ulike servere.

Forkunnskaper

Prosjektgruppen har jobbet med JavaScript tidligere og har relativ god forståelse av språket. Gruppen har derimot ikke jobbet med rammeverket Node.js før.

3.1.2.3 Java

Java er et programmeringsspråk som er klassebasert, objekt-orientert og designet for å ha få avhengigheter (Oracle, u.å.). Måten vi tenkte å lage en webapplikasjon i Java på, er ved hjelp av Java Servlets. Java Servlets er en veldig underliggende teknologi som blir benyttet av blant annet Struts.

Forkunnskaper

Alle i prosjektgruppen har jobbet mye med Java tidligere og er veldig gode i og trygge på dette programmeringsspråket. I tidligere fag har vi jobbet mye med Java Servlets og vi har god forståelse for hvordan de ulike komponentene fungerer og kommuniserer.

3.1.3 Vurdering og diskusjon

For dette prosjektet har vi begrenset med tid, men også et ønske om å levere noe etter de satte kravene fra oppdragsgiver. Tidsmessig ville et forbedret Excel-ark vært det mest praktiske, men det oppfyller ikke kravspesifikasjonene. Vi foretrekker derfor å utvikle et produkt som krever mer tid og innsats av oss, men som til gjengjeld blir på ønsket format.

Oppdragsgiver har også en liste over ønskelige funksjoner. Mange av disse funksjonene vil ikke være realistisk for oss å få tid til å ordne, men vi ønsker å legge opp produktet på en slik måte at det vil være mulig for bedriften å videreutvikle det senere.

Når det kommer til hvilket programmeringsspråk og rammeverk vi ønsker å benytte oss av, så finnes det både fordeler og ulemper med alle.

Vi har mye kunnskap rundt Java Servlets, og vet hvor mye arbeid som må legges inn for å gjøre små ting. Vi ønsker å implementere mange funksjoner i webapplikasjonen vår, noe som kan bli veldig tidkrevende å gjøre i Java Servlets.

Node.js har de funksjonene vi trenger for å kunne lage webapplikasjonen vi ønsker, med høy hastighet og mange muligheter til å legge til ulike funksjoner. Man kan lage et flott brukergrensesnitt hvor man legger objekter inn i lister i sanntid. Node.js kan derimot være vanskelig å vedlikeholde siden man ofte integrerer mange ulike APIer i applikasjonen som kan kreve oppdateringer.

Python Flask har alle de funksjonene vi trenger for en webapplikasjon på lik linje med Node.js. Vår tekniske veileder på SGG har god kompetanse innenfor Flask noe som gjør at vi kan få hjelp når vi trenger det. I tillegg er Flask kjent for å være enkelt å lære seg og at man raskt kan sette opp et fungerende produkt. Det at vi raskt kan få opp en prototype kan igjen gjøre det enklere for oss å holde god kommunikasjon med oppdragsgiver når det kommer til definering av kravspesifikasjoner til produktet.

Grunnet gode ressurser rundt rammeverket Flask ønsker vi å bruke dette til utviklingen av webapplikasjonen.

3.2 Spesifikasjoner

Vi valgte løsningen med å lage en webapplikasjon. Grunnen til at vi landet på dette valget var hovedsakelig oppdragsgivers krav til løsningen. Under punkt 3.1.2 definerte vi både fordeler og ulemper med denne løsningen, og til tross for et par ulemper var dette løsningen som svarte best til det oppdragsgiver ønsket.

Med alle de ønskelige funksjonene og funksjoner som absolutt måtte med, var det i grunn aldri et alternativ å lage en forbedret versjon av Excel-arket. De nåværende skaleringsproblemene og problemene med hvor lite oversiktlig det var ville fulgt løsningen videre.

3.3 Valg av verktøy, programmeringsspråk og rammeverk

Python Flask

Flask er et mikrorammeverk for Python basert på Wekzeug og Jinja (Flask, u.å.). Det er veldig lett å komme fort i gang med Flask, noe som gjør at vi kommer fort i gang med oppgaven. Det var også et ønske fra oppdragsgiver at vi skulle bruke dette for lett integrasjon med deres intranett, og det var derfor et enkelt valg for oss å velge dette.

SQLAlchemy

SQLAlchemy er et Python SQL verktøysett som gir applikasjonsutviklere kraften og fleksibiliteten til SQL (SQLAlchemy, u.å.).

PostgreSQL

Databasen som SGG allerede har er skrevet med PostgreSQL - derfor ble dette også et enkelt valg for oss når vi skulle velge databasesystem.

Microsoft Teams

Vi bruker Microsoft Teams for lagring av dokumenter og planlegging i gruppen. Enkelte av gruppemedlemmene hadde erfaring med dette verktøyet fra tidligere og var fornøyd med måten det fungerte på.

Slack

For enkel kommunikasjon med Stian og Helene valgte vi å bruke Slack. Det er et kommunikasjonssystem som brukes hos flere bedrifter og kan brukes til mindre oppdrag. Slack var allerede det foretrukne kommunikasjonssystemet hos bedriften og den enkleste måten å kontakte oppdragsgiver på.

3.4 Utviklingsmetode for prosjekt

3.4.1 Utviklingsmetode

Utviklingen skal foregå i iterasjoner. For dette prosjektet har vi landet på et rammeverk som heter Scrum. Scrum er et rammeverk som hjelper team å jobbe sammen på en strukturert og effektiv måte (Scrum, u.å). Det blir ofte omtalt som et smidig agilt rammeverk for prosjektledelse, noe som vil si at det er basert på iterativ utvikling. En slik utviklingsmetode åpner for å avdekke svakheter i programmet på et tidlig stadium og kontinuerlig forbedre produktet underveis.

3.4.2 Prosjektplan

Da valget av fremgangsmåte var gjort ble det foretatt et estimat av hvor lang tid hver fase i prosjektet ville ta. Det ble tatt hensyn til manglende kunnskap ved utformingen av planen for prosjektet.

Prosjektplanen er laget i form av et GANTT-diagram. Hver fase er beskrevet med små sprinter, hvor hver sprint er på to uker. Fullstendig diagram finnes under punkt 10.2. Den består av de fem fasene:

1. Planlegging
2. Design og arkitektur
3. Implementering
4. Validering
5. Jobbe med rapport

Sistnevnte vil foregå gjennom hele utviklingsperioden og er i seg selv ikke en egen fase. Målet med å ha en slik inndeling er bevisstgjøring av hva som er nåværende fokus og for vurdering av fremdriften i prosjektet.

Diagrammet vil fungere som en veileder underveis for å gi en pekepinn på hvordan vi ligger an og om vi eventuelt er nødt til å justere kursen og endre fokuset. Diagrammet er basert på de ulike kravene fra oppdragsgiver og fastsatte frister fra skolen.

Gruppen har også utnevnt en prosjektleder, Anders, som er ansvarlig for arbeidsfordelingen og å sikre prosjektets fremgang. Mer spesifikt vil dette si at han har ansvar for at gruppen overholder frister som er satt og at han til enhver tid har oversikt over fremgangen i forhold til prosjektplanen.

I realiteten vil ikke de ulike rollene ha særlig innvirkning på hvordan prosjektet utføres, siden vi legger stort fokus på samarbeid og ansvarlighet fra hvert enkelt gruppemedlem.

3.4.3 Risikohåndtering

For å kunne håndtere ulike risikoer var det nødvendig å identifisere dem. Etter at ulike risikoer var identifisert ble det utarbeidet en risikoliste. Denne listen er definert under 10.1: Risikoliste, og har tilhørende reaktive og proaktive tiltak definert. De proaktive tiltakene er tiltak som kan iverksettes for å unngå at en eventuell risiko utspiller seg, mens de reaktive tiltakene er tiltak som skal iverksettes dersom prosjektet allerede er rammet.

Å håndtere risikoer handler om å minimere en eventuell innvirkning det vil ha på prosjektet. Risikolisten har en sannsynlighets- og alvorlighetsgrad for hver forekomst og gir dermed et klart uttrykk for hvilke risikoer man bør være ekstra var på. Dersom en risiko har både høy sannsynlighets- og alvorlighetsgrad er det særlig viktig at vi har gode rutiner for behandling av en eventuell innvirkning av den.

I dette punktet vil de ulike risikoene bli beskrevet mer detaljert.

Sykdom

Sykdom er utenfor vår kontroll, men det finnes tiltak vi kan gjøre for å minske sannsynligheten for å bli syk og eventuelt korte ned sykdomsforløpet – blant annet å passe på at alle gruppemedlemmer får nok pauser og å unngå overarbeid.

Når uhellet først er ute er det viktig å kommunisere et eventuelt fravær til de andre gruppemedlemmene snarest mulig. Det vil dermed være mulig å fordele arbeidsoppgavene annerledes slik at målene nås i tide. Ved kortvarig sykdom, som det i de fleste tilfeller vil være, vil ikke innvirkningen være særlig stor. Det kreves derimot at de gjenværende gruppemedlemmene legger inn en større innsats over sykdoms perioden.

Dårlig samarbeid innad i gruppen

Innvirkningen av et dårlig gruppesamarbeid vil være svært alvorlig. Det er derimot ikke særlig sannsynlig, da gruppen på forhånd har blitt enige om arbeidsmengde, oppmøtedager og er enige om ønsket resultat. Det har også vært god kommunikasjon fra starten av.

Et dårlig samarbeid kan skape grobunn for et dårlig arbeidsmiljø og vil ikke bare påvirke hvor mye gruppen som helhet får gjort, men også motivasjonen og stemningen til hver enkelt.

En mulig reaktiv løsning er å avholde et møte hvor alle parter får ytret det de måtte ønske og på den måten forhåpentligvis komme frem til en felles løsning.

Dårlig kommunikasjon med oppdragsgiver

Helt fra starten av har et tett og godt samarbeid med oppdragsgiver vært vektlagt. Hele arbeidsprosessen foregår i SGG sine lokaler, og oppdragsgiver er stort sett tilgjengelig dersom noe er uklart.

Proaktive tiltak som gjøres for å unngå misforståelser er å jevnlig oppdatere både Stian og Helene på hva vi tenker, hvordan vi planlegger å løse de neste stegene og å passe på å få tilbakemeldinger for å eventuelt justere kursen.

Skjevfordeling av arbeid

Skjevfordeling av arbeid henger sammen med punktet om dårlig samarbeid innad i gruppen. Proaktive tiltak som gjøres for å unngå dette er å fordele arbeid etter evner og gi hvert enkelt gruppemedlem oppgaver de har mulighet for å klare å løse. Dermed unngår vi at gruppemedlemmer blir stående fast og dermed ikke får gjort den arbeidsmengden som er satt.

Misforståelser knyttet til kravspesifikasjonene

Misforståelser knyttet til kravspesifikasjonene henger sammen med punktet om dårlig kommunikasjon med oppdragsgiver. Det er viktig å stille spørsmål dersom noe er uklart; dette gjelder også ved god kommunikasjon. Her er det viktig å være bevisst på hvordan egen oppfattelse ikke nødvendigvis stemmer overens med hva den andre parten prøver å kommunisere.

Dårlig tidsplanlegging

Før prosjektet startet ble det utarbeidet en prosjektplan i form av et GANTT-diagram. Dersom tidsfrister ikke overholdes her er det fort gjort å falle bakpå i forhold til planen. Konsekvensen av dette kan være at 1. et ufullstendig prosjekt må leveres eller 2. et fullstendig, men dårlig utført, prosjekt må leveres. For å unngå dette vil det jevnlig gjøres sjekker på hvordan vi ligger an i forhold til diagrammet og om fokuset vårt ligger på riktig del av prosjektet.

Utstyr som ikke fungerer

Gruppemedlemmene stiller selv med PC og eventuelt tilleggsutstyr de ønsker å bruke. Bedriften har tilgjengeliggjort annet utstyr som skjerm, tastatur, diverse kabler og kontorlokaler. Dersom en PC skulle slutte å fungere vil vi alltid ha en sikkerhetskopi av prosjektet på en annen PC eller Microsoft Teams / GitHub, og det vil dermed ikke være særlig kritisk. I et slikt tilfelle er den reaktive løsningen å finne en annen PC som man kan bruke midlertidig.

Manglende kompetanse

Prosjektet skulle utføres ved hjelp av språk og teknologier som gruppen ikke var kjent med fra før av. Løsningen her ble å sette av tid i starten av prosjektet til å gå gjennom ulike tutorials for å få grunnforståelsen på plass.

Når ikke satte mål

Dersom vi ikke når de satte målene er det sannsynligvis som et resultat av overnevnte risikoer. Om vi ikke når satte mål er det viktig å jobbe videre for å prøve å nå neste mål. På den måten vil vi øke sjansene for innhenting av arbeidet. Dette gjelder ikke dersom neste mål er fullstendig avhengig av at vi når det nåværende målet. Innvirkningen av at vi ikke når satte mål avhenger av hvorfor vi ikke har nådd de. Små misforståelser vil være enklere å fikse enn et dårlig samarbeid i gruppen.

3.5 Evalueringsmetode

For å finne ut om resultatet har oppnådd målet er det ønskelig å la oppdragsgiver, Stian, prøve ut produktet over en periode. Vi vil da kunne se om produktet fungerer slik det var ønsket og om oppsettet er intuitivt og enkelt å bruke. En slik evaluering over tid åpner for direkte tilbakemelding om hvorvidt prosjektet var en suksess og gir forhåpentligvis gruppen nok informasjon til å kunne evaluere sluttresultatet.

Denne evalueringen vil finne sted under den siste fasen av prosjektet. Da skal alt av funksjonalitet være på plass og vi kan dermed presentere et helhetlig produkt som skal utprøves.

Det vil også være en løpende evaluering underveis gjennom hele prosjektet. Siden mesteparten av arbeidet vil bli gjort i SGG sine lokaler er det enkelt for Stian å komme med tilbakemeldinger etterhvert som vi implementerer nye funksjoner. Vi får dermed verdifull informasjon løpende som kan brukes til å evaluere hver utviklingsfase. Her er det også mulig for oss å forhøre oss med Helene for å få tilbakemeldinger på de mer tekniske aspektene.

I rapporten, under punkt 5.1, har vi definert de ulike evalueringsmetodene vi vil benytte oss av for å finne ut om resultatet har oppnådd målet.

Det vi da ønsker å sjekke er følgende punkter:

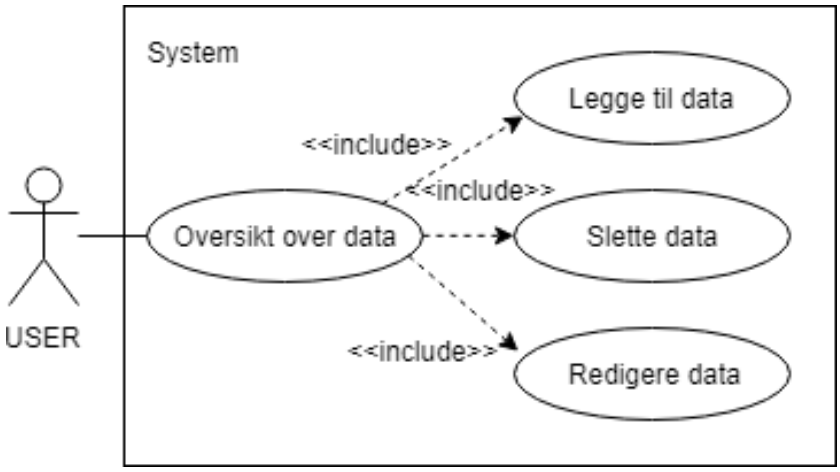
1. Det er mulig å administrere, det vil si slette, legge til og endre både ansatte, transaksjoner, kvitteringer og utstyr
2. Alle ønskelige advarsler kommer opp og gir riktig informasjon
    a. Ansatte som har sluttet og fremdeles har registrert utstyr på seg
    b. Utstyr som ikke er registrert på noen ansatte
    c. Utstyr som mangler kvittering
    d. Utstyr med garantitid som snart løper ut
3. Webapplikasjonen er intuitiv og enkel å bruke

De overnevnte punktene vil sjekkes under både den løpende evalueringen og under sluttevalueringen.

4 Produktdesign

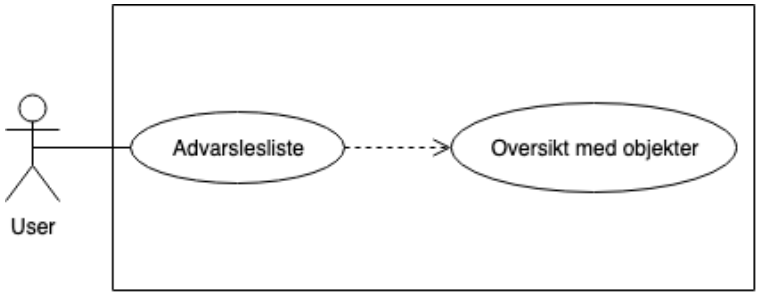
I denne delen vil vi presentere de underliggende designprinsippene og applikasjonens arkitektur. Vi har valgt å følge MVC (Model View Controller) som er et designmønster for utviklere. Dette designmønsteret legger til rette for gjenbrukbar kode og parallell utvikling.

4.1 Brukstilfeller



Figur 4.1: brukstilfelle for oversiktssidene

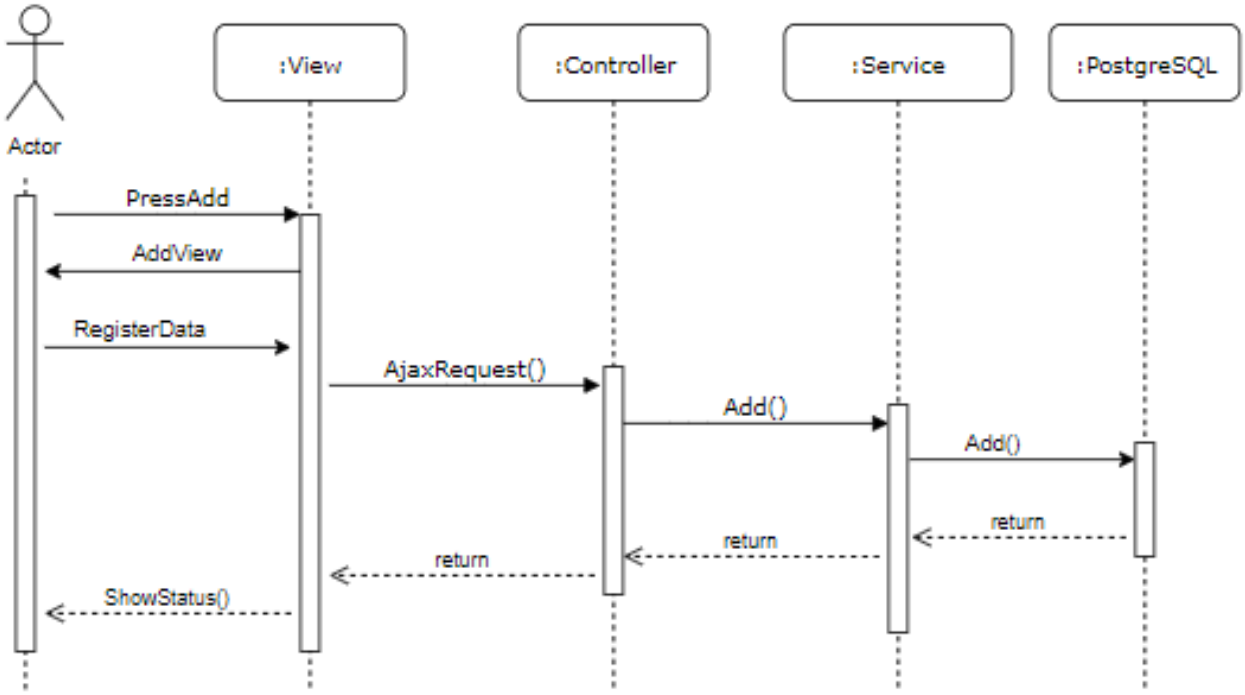
Det første brukstilfelle viser hva alle oversiktssidene skal støtte. Brukeren skal kunne se en oversikt over all data som ligger inne i databasen. I denne oversikten vil det være mulig å legge til ny data, slette data og redigere data - alt på samme side.



Figur 4.2: brukstilfelle for advarselside

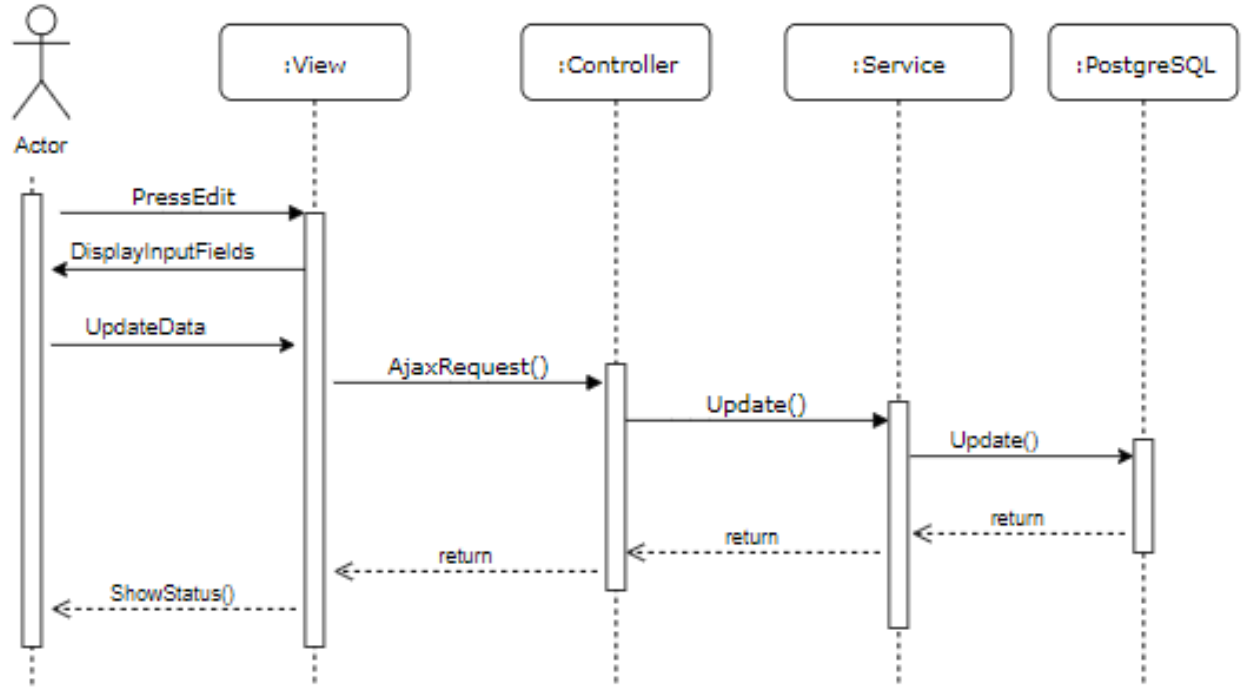
Det neste brukstilfellet, vist i figur 4.2, gir en visuell beskrivelse av hvor enkelt det skal være å bruke advarslene som er implementert i applikasjonen. Når noen objekter har en tom eller ugyldig verdi, vil det komme opp en advarsel på advarselssiden. Ved trykk på advarselen skal brukeren komme til en oversikt over de gjeldende objektene. Fra denne oversikten kan også alle de samme funksjonene som er vist i figur 4.1 brukes.
Videre i dette avsnittet vises brukstilfeller på hvordan de forskjellige delene av applikasjonen henger sammen og kommuniserer, samt hvilke metoder som kalles.

4.1.1 Legge til ny ansatt/utstyr/kvittering



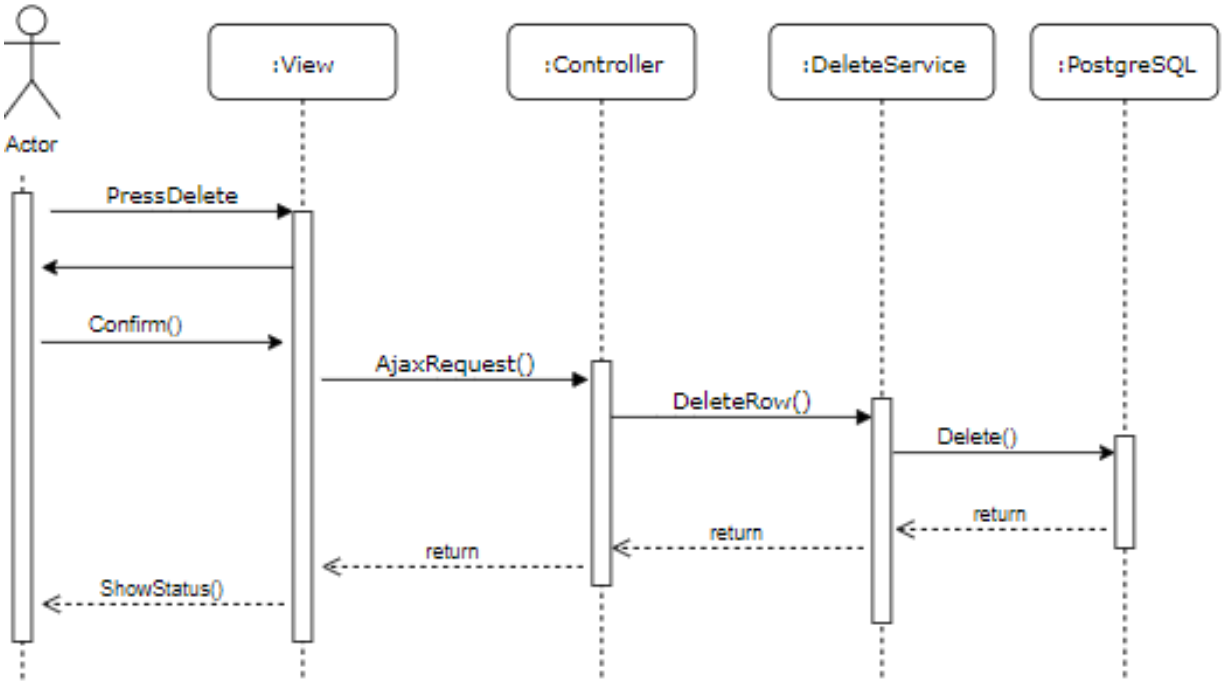
Figur 4.3: Sekvensdiagram for å legge til et nytt objekt

4.1.2 Endre på data som ligger inne



Figur 4.4: Sekvensdiagram for å endre på data

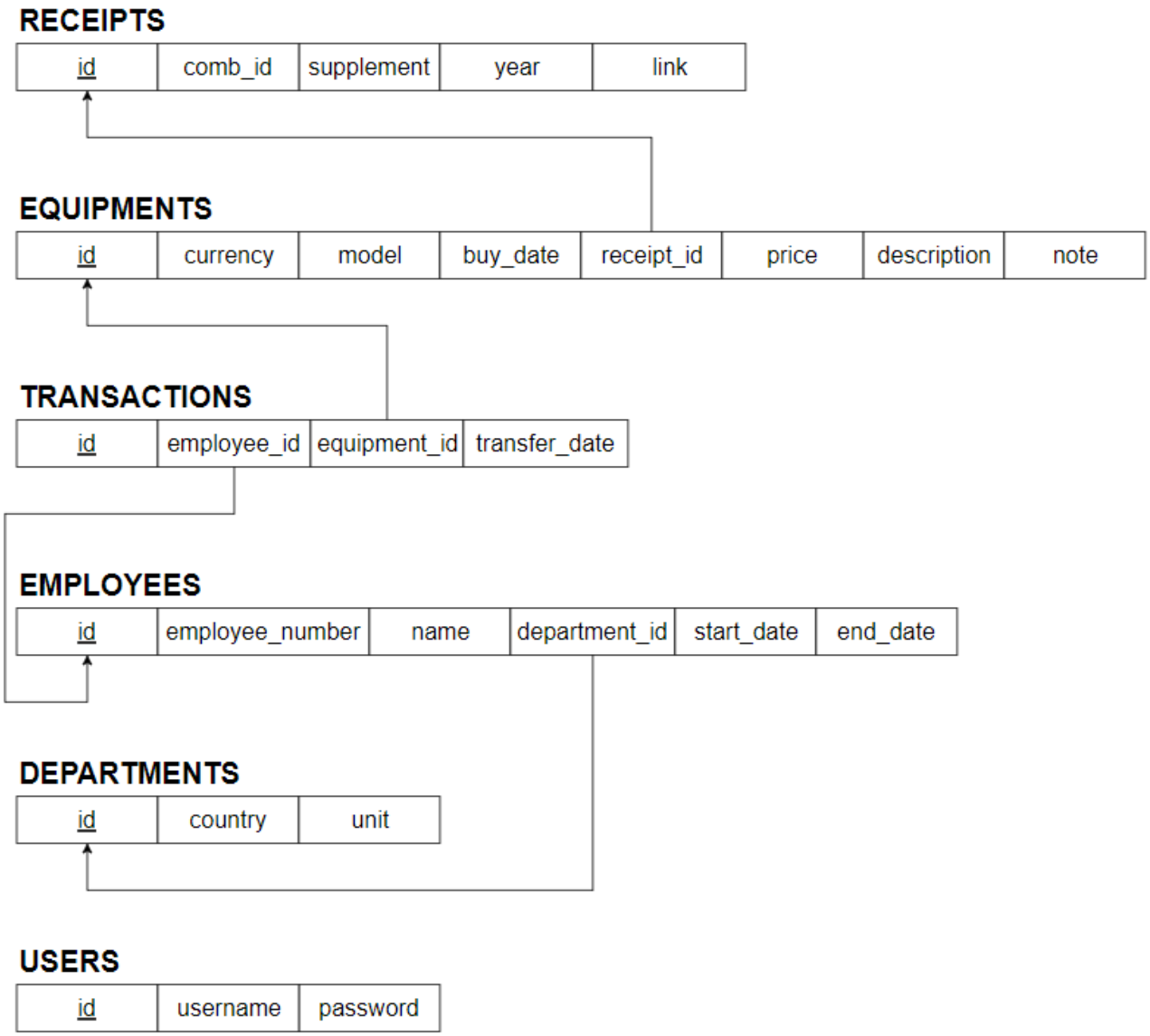
4.1.3 Slette en rad



Figur 4.5: Sekvensdiagram for å slette en rad

4.2 Database

Databasen var det første vi implementerte i løsningen. Som beskrevet i punkt 3.3 bruker vi en PostgreSQL database. Det var viktig å etablere databasen tidlig, da databasen er den underliggende delen i hele applikasjonen. Alle tjenestene på serveren bygger på tabellene databasen, som igjen blir brukt for å representere ulike views vi har i applikasjonen (se punkt 4.3.4). På grunn av denne avhengigheten var det sentralt å få gjort databasen ferdig tidlig. Det ble satt av en hel uke til planlegging av databasen, og vi fikk raskt laget en lokal database som var i stand til å registrere og lagre data som oppdragsgiver hadde gitt oss. Denne databasen har kontinuerlig blitt endret underveis i prosjektet etterhvert som det kom nye funksjoner vi ønsket å implementere, som ikke nødvendigvis var planlagt fra starten. Det å gjøre endringer i databasen mot sluttfasen av prosjektet har vært både tidkrevende og vanskelig, men det har også vært uunnværlig og en naturlig del av utviklingsprosessen.



Figur 4.6: Relasjonsdiagram for databasen

Databasen er delt inn i to deler. Den ene delen tar seg av alt som har med bedriftsressursene å gjøre, og den andre delen er brukere som har tilgang på dataen.

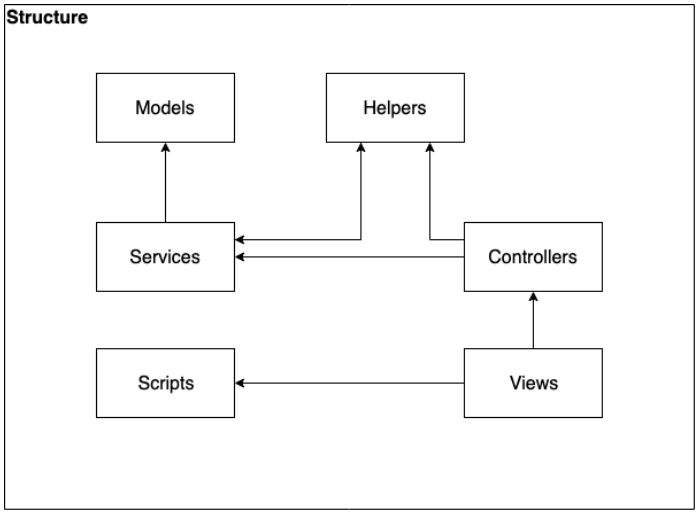
Hoveddelen av databasen er delt inn i 5 deler; RECEIPTS, EQUIPMENTS, TRANSACTIONS, EMPLOYEES og DEPARTMENTS. Hver enkelt tabell har en id som blir automatisk generert. Denne delen av databasen inneholder all informasjonen som kan legges til og redigeres i programmet.

I databasen har vi også laget tabellen USERS. Denne tabellen blir brukt til innlogging i applikasjonen. Den består av id, brukernavn og passord. Vi ønsker ikke at passordet skal bli lagret i leselig tekst, så vi har valgt å kryptere dette før det blir sendt til selve databasen.

4.3 Webserver

Vi har valgt å følge Model-View-Controller (MVC) mønsteret for å bygge opp webapplikasjonen vår. Som tidligere beskrevet bruker vi mikrorammeverket Flask i Python til å bygge opp applikasjonen vår, noe som ikke er et MVC rammeverk. Vi har likevel valgt å strukturere programmet vårt som et MVC program. Det er flere fordeler med å følge MVC: en av de er at det er lett å jobbe på flere deler av prosjektet samtidig. Hvis en person jobber med model kan en annen person jobbe parallelt med det som er i view. En slik arbeidsmetode fører til raskere og mer effektiv utvikling av applikasjonen. Andre ting som er nyttig med MVC er at det separerer funksjonalitet og objekter, noe som gjør at programmet blir mer robust og gjør at endringer ikke påvirker hele modellen. Endringer blir mindre tidkrevende og det fører til færre feil i programmet når de først må gjøres.
MVC gjør at koden kan lett brukes over flere forskjellige kodespråk, siden objektene som sendes alltid er de samme.

Vi vil nå diskutere de forskjellige delene av programmet og hvordan de henger sammen. Figur 4.7 viser en nedskalert oversikt over hvordan de forskjellige delene av programmet henger sammen. Vedlegg 10.7 viser et klassediagram over hele applikasjonen og gir en oversikt over hvordan arkitekturen til applikasjonen er bygget opp.



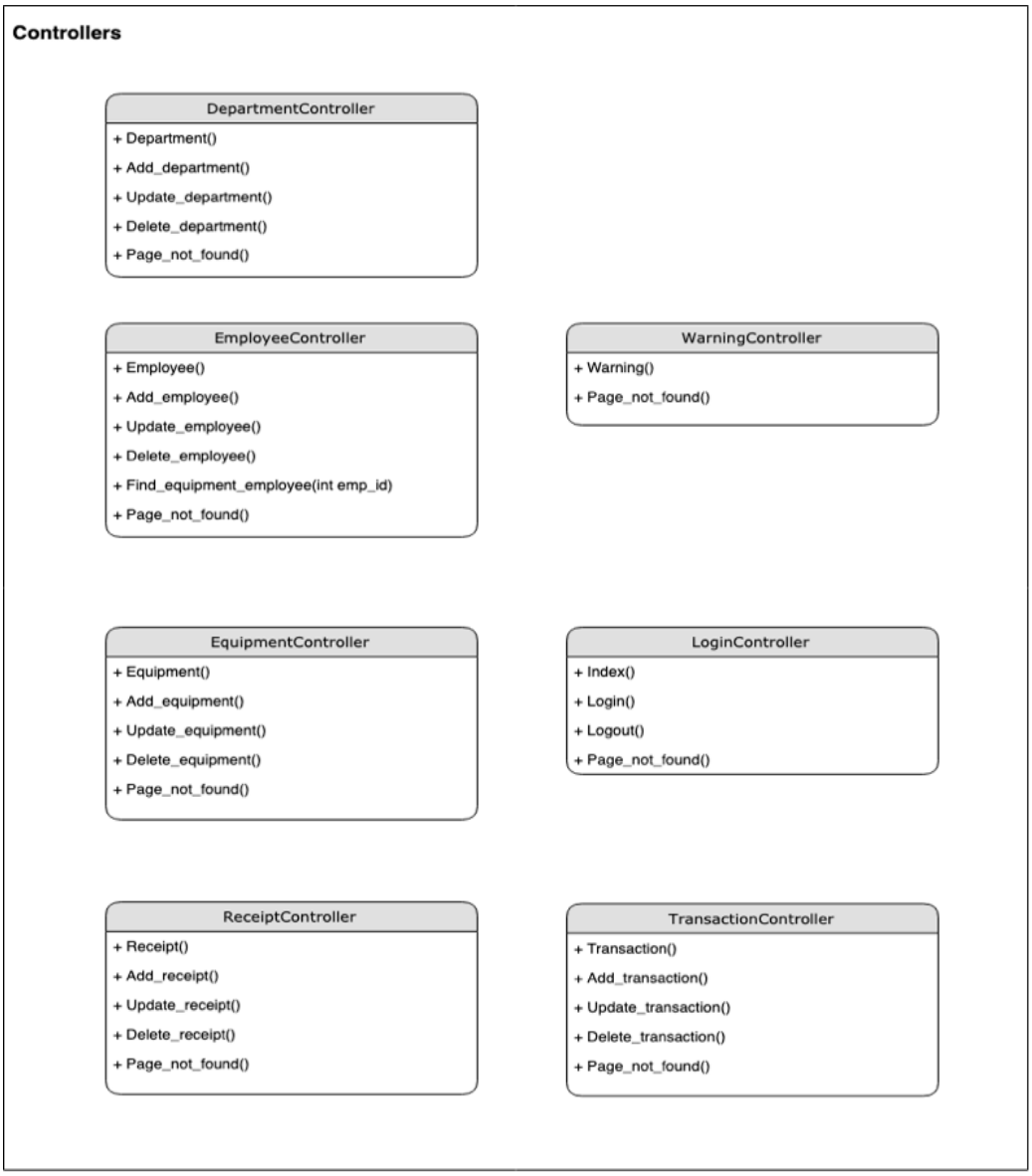
Figur 4.7: Oversikt over applikasjonsarkitekturen

4.3.1 Controller

«Controller - Controller acts on both model and view. It controls the data flow into model object and updates the view whenever data changes. It keeps view and model separate. » (Design Patterns - MVC Pattern, u.å.).

Controller er den delen av applikasjonen som styrer logikken som skjer i view, beskrevet i punkt 4.3.4. Controller separerer model og view, legger til ny data i databasen og oppdaterer view når dataen i databasen endres. Controller fungerer som et mellomledd fra logikken som skjer frontend og all logikken som brukeren ikke ser backend.

Controller håndterer all data som sendes til og fra view. Controller setter opp hvordan lenken skal se ut og hvilket view som skal lastes på den lenken. Figur 4.8 viser en oversikt over controller-klassene som er med i applikasjonen, samt hvilke metoder hver av de enkelte klassene har.

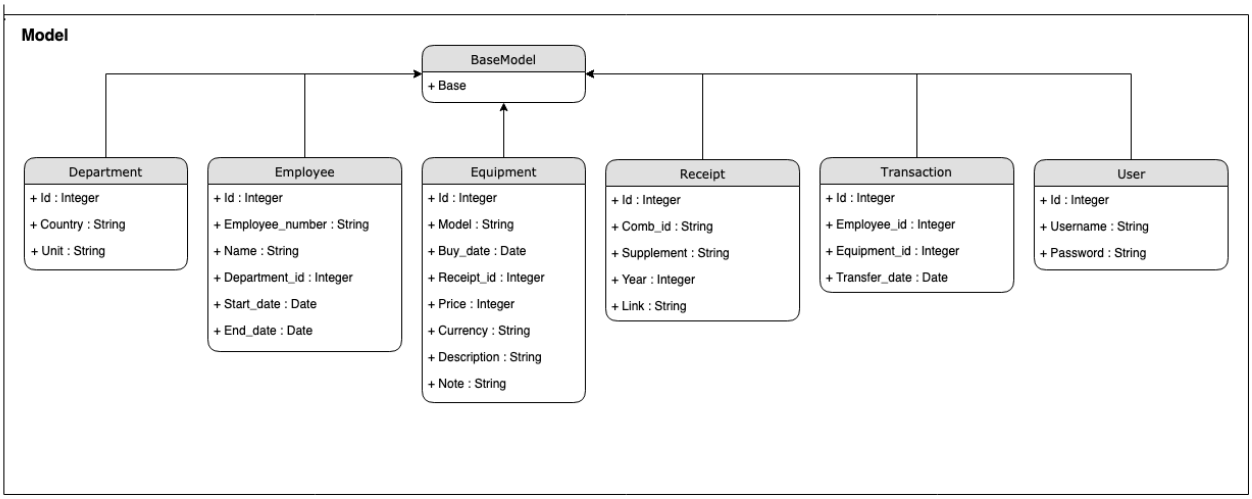


Figur 4.8: Oversikt over controller-klassene

Når controllene har fått data fra view sendes dataene videre til hjelpemetodene sine i service. På denne måten går dataene nedover i hierarkiet. Bruker skriver inn i view, view sender dataene videre til controller, controller gjør om på noe av dataene og sender det til service. Deretter sender Servicene det til databasen og oppdaterer den.

4.3.2 Models

Modellene er klasser som representerer objekter i databasen. Det finnes en klasse for hver tabell som ligger i databasen.



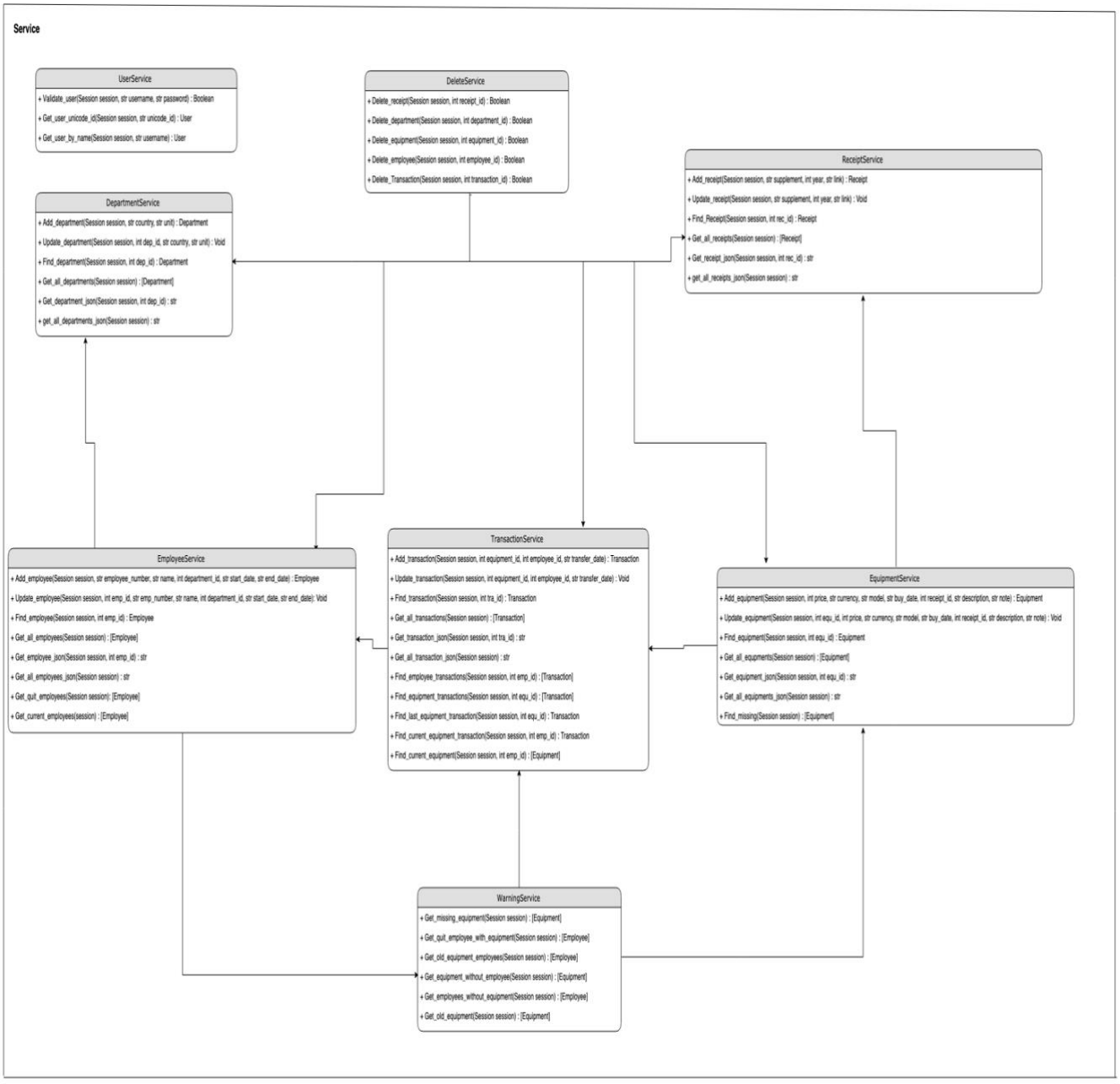
Figur 4.9: Oversikt over modellene

Figur 4.9 viser en oversikt over hvilke modeller vi har. Alle modellene arver fra BaseModel som er en modell som brukes av Flask rammeverket for å bygge opp modeller og sammenkoble dem med en database.

4.3.3 Services

Services er hjelpeklasser til controller. All logikk som ligger i disse klassene kunne ha vært i controller, men vi valgte å legge logikken i disse klassene da det gir enklere og mer oversiktlig kode. Alle databasespørringer skjer i disse klassene. Hver modell har sin egen service for å kunne håndtere de forskjellige logiske handlingene. Handlingene vi har implementert inkluderer å legge til en ny instans av denne modellen i databasen, oppdatere data som allerede finnes i databasen, finne et objekt fra en modell lagret med en identifikator, og få en liste med all data av en modell som finnes i databasen.

Det er også implementert services for å håndtere andre ting, som å finne advarsler for data som er lagt inn. Disse advarslene viser for eksempel om et utstyr som er lagt til i listen mangler noe data eller om noen som har sluttet i bedriften fortsatt har utstyr tildelt til seg selv.

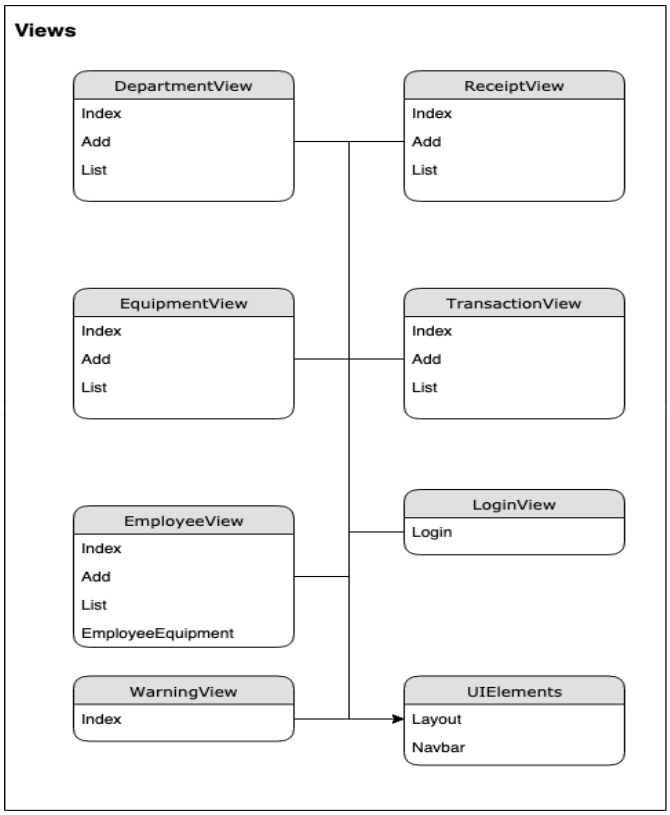


Figur 4.10: Klassediagram over services

4.3.4 Views

Views inneholder all htmlkoden som skal vises til brukeren. Her har vi brukt Jinja2 til å koble sammen html filer til å bli vist på samme side. Alle sidene bruker samme navigasjonsbar på toppen, og alle bruker et standard oppsett av html5 som vi har definert i en layout fil. En slik sammenkobling av filer gjør at det blir mindre kode, da man slipper å skrive det samme om igjen og heller kan benytte oss av de samme filene på flere steder. Hvert view har en egen controller som skal sørge for at det er riktig data som blir vist. Den samme controlleren skal også håndtere data som brukeren sender inn via views. Figur 4.11 viser en oversikt over mappestrukturen til jinja2 filene som er brukt, og hvilke filer de inneholder. Figuren viser hvordan alle views henter

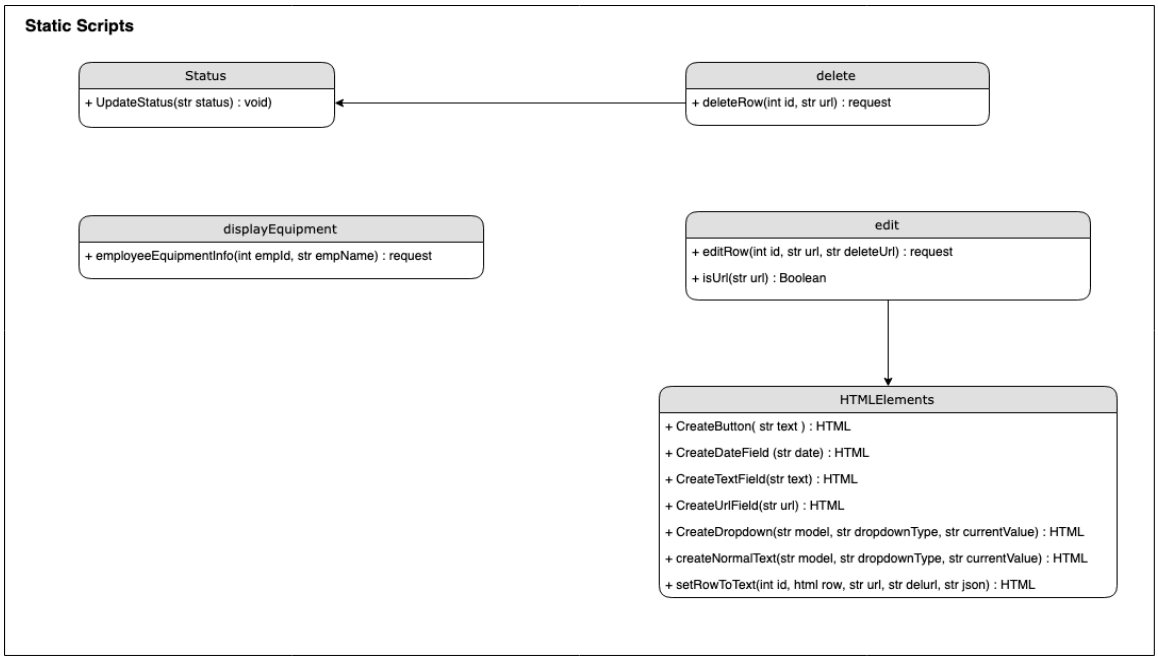
layout og navbar fra UIElements. Navbar er navigasjonsbaren som er øverst på hver side og layout er et enkelt htmloppsett som alle sidene bruker. Den eneste siden som ikke henter navigasjonbaren er LoginView. LoginView skal, av sikkerhetsgrunner, ikke hente navigasjonsbaren.



Figur 4.11: Jinja2 filer mappestruktur

4.3.5 Scripts

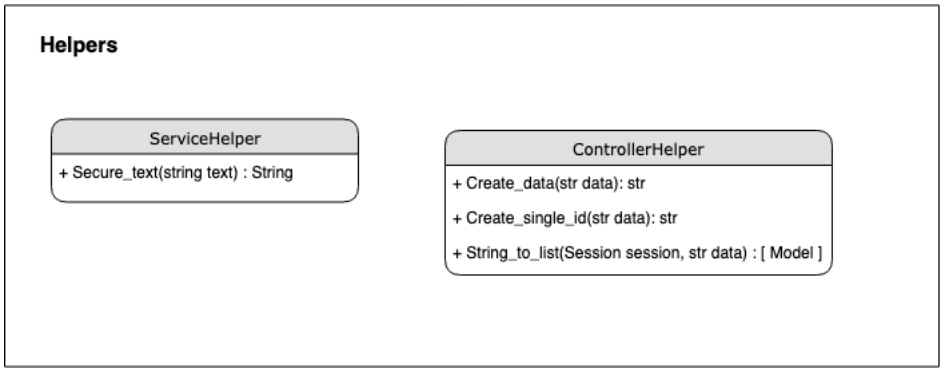
All JavaScript-koden befinner seg i scripts. Denne koden virker som en hjelper til views, da det ikke er all logikk som lett kan bli håndtert i controllerne. Det som lettere kan håndteres i JavaScript blir heller håndtert i det. Man håndterer gjerne ting som endrer på htmlkoden som allerede ligger i views. Edit funksjonen, som endrer hvordan htmlkoden i tabellen ser ut, er et eksempel på en av funksjonalitetene som er lettere å få til med JavaScript enn med Python. Når siden dynamisk skal vise om et objekt er blitt slettet eller om det er lagt til, brukes det JavaScript.



Figur 4.12: Script oversikt

4.3.6 Helpers

Det er to ekstra hjelpeklasser som blir brukt i denne applikasjonen. Disse hjelpeklassene er ServiceHelper og ControllerHelper. Som navnene tilsier er de hjelpere til services og controller. Disse klassene inkluderer metoder som flere av servicene eller controllerne bruker, og det er derfor blitt valgt å definere dem i egne klasser.



Figur 4.13: hjelpeklasser med metoder

4.4 Klient

Klientsiden av applikasjonen ble laget ved hjelp av programmeringsspråkene JavaScript, HTML og CSS, og ved hjelp av verktøyene Jinja2 og Bootstrap. Bootstrap er en verktøykasse med åpen kildekode som kan brukes for rask utvikling med HTML, CSS og JS (Bootstrap, u.å.).

4.4.1 HTML, Jinja2 og Bootstrap

Prosjektet består av totalt 7 HTML-sider. Alle disse sidene, med unntak av innloggingssiden, har en relativt lik struktur. De har et enkelt skjelett, en navigasjonsbar og en liste av objekter. Jinja2 gjør det enkelt å holde en god struktur i HTML-dokumentene. I prosjektet har vi en navigasjonsbar i toppen av alle HTML-filene, samt en basis HTML-layout hvor vi definerer header, body og alle importeringer. Ved hjelp av Jinja2 kan vi definere disse én gang og bruke de i alle de ulike HTML filene vi har. Vi kan også definere alle HTML-objektene i separate filer og deretter sette de sammen.

```
<div class="row justify-content-md-center">
    <div class="col">
        {% include 'Views/Equipment/list.html' %}
    </div>
</div>
{% include 'Views/Equipment/add.html' %}
```

Figur 4.14: Utdrag fra koden som viser hvordan Jinja2 blir brukt

Som vi kan se i figur 4.14 er det enkelt å hente inn kode fra andre plasser i prosjektet. Vi har også brukt Jinja2 til å gjøre logiske operasjoner i genereringen av HTML-filene. Prosjektet er i stor grad basert på å lage lister og å bruke metoder for å representere disse.

```
<select id="employee" name="employee">
    {% for emp in employee_data %}
        <option value="{{ emp.id }}">{{ emp.name }}</option>
    {% endfor %}
</select>
```

Figur 4.15: Bildet viser hvordan vi enkelt kan lage en nedtrekksmeny

Ved hjelp av Jinja2 kunne vi lage logiske utrykk for å iterere gjennom lister og gjøre enkle spørringer, noe som ble brukt mye gjennom hele prosjektet. Dette verktøyet har vært utrolig nyttig for prosjektet. Jinja2 er veldig stort, med mange ulike funksjoner, men med dokumentasjon som har vært vanskelig å bruke. Dette kan ha ført til at verktøyet ikke ble utnyttet så godt som det kunne ha blitt.

For å gjøre webapplikasjonen estetisk vakker, har vi valgt å bruke Bootstrap som er et frontend-rammeverk. Med dette rammeverket kan vi definere hvilket utseende vi ønsker på de ulike objektene direkte i klassenavnene deres. En slik funksjonalitet har vært spesielt nyttig i tilfeller

hvor vi ønsket å lage visuelt fine knapper og pop-up-menyer for registrering og endring av objekter.

For eksempel har det vært mulig for oss å skrive:

```
<nav class="navbar navbar-static-top navbar-expand-md navbar-dark bg-dark">
```

Og dermed har vi kunnet definere på én linje både hvor vi ønsker denne navigasjonsmenyen og hvordan vi ønsker at utseendet på den skal være.

4.4.2 JavaScript

I prosjektet har vi brukt JavaScript for å gjøre sidene dynamiske. Det ble ikke brukt noe rammeverk for dette, men heller ren JavaScript. For å gjøre siden enkel å bruke er det sentralt at minst mulig informasjon blir vist på en gang. Muligheten til å endre, legge til og slette er lagt til på samme side, enten i en pop-up-meny eller direkte i raden i tabellen. Vi har brukt metoder som Ajax for å sende metodekall asynkront til webserveren. På denne måten føles applikasjonen rask og det er ikke nødvendig å laste ned hele siden på nytt når endringer blir gjort.

4.5 Sikkerhet

Denne applikasjonen er laget for internt bruk i SGG, og det er derfor ikke lagt mye fokus på sikkerhet i applikasjonen. Selv om det ikke har vært mye fokus på det, så er applikasjonen sikker mot de vanligste angrepene. Den er sikker mot HTML-injection, noe som vil si at hvis brukeren skriver inn ren tekst i form av HTML kode så skal det ikke kjøres som HTML kode. Applikasjonen oversetter derfor hele teksten slik at den ser lik ut for brukeren, men den er ikke lenger et problem. Helpers, se 4.3.6, har metoder for å oversette fra ren tekst for services og controller. Det er implementert en innloggingsside for at ikke hvem som helst som er på internettet til SGG skal kunne komme seg inn og lese all informasjonen som ligger i applikasjonen.

5 Evalueringer

Hovedmålet med prosjektet var å lage en webapplikasjon som ville forenkle hverdagen til oppdragsgiver.

For å evaluere om prosjektgruppen hadde klart å oppnå dette målet ble det gjort ulike evalueringer for å teste i hvilken grad dette var gjort.

5.1 Evalueringsmetoder

5.1.1 Intervju med oppdragsgiver

Ved ferdigstilling av produktet er det ønskelig å avholde et intervju med oppdragsgiver for å avdekke hvorvidt produktet dekker de krav som er satt og i hvilken grad det er intuitivt å bruke. Underveis i prosjektet vil det gjennomføres samtaler med oppdragsgiver for å få tilbakemeldinger, så sluttintervjuet vil dermed sannsynligvis ikke avdekke store mangler eller feil ettersom disse skal ha blitt korrigert underveis.

Dette vil derimot være første gang oppdragsgiver får testet ut det ferdige produktet og testet ut alle funksjoner på egenhånd. Ved tidligere anledninger har prosjektgruppen gitt en kort gjennomgang av nye funksjoner og endringer etterhvert som de ble lagt til. Gruppen har da selv vist frem webapplikasjonen, og dermed har det ikke vært mulig å få reelle tilbakemeldinger på hvorvidt det er intuitivt og enkelt å bruke eller ikke.

Under intervjuet skal følgende spørsmål bli besvart:

- Om funksjonaliteten i programmet står til kravspesifikasjonene
- Om det er deler av programmet som ikke har særlig nytteverdi
- Om det er en plan videre, fra bedriften sin side, for prosjektet
- Om programmet er enkelt og oversiktlig å bruke
    - Eventuelt hva som kunne forbedret programmet på disse punktene
- Om noe av funksjonalitet, utover kravspesifikasjonene, er savnet
- Hvem som skal benytte seg av programmet

Prosjektgruppen vil selv ha gått gjennom kravspesifikasjonene på forhånd og sjekket at ting er på plass, men et slikt intervju åpner for mer subjektive synspunkt og gir innsikt i ting vi som prosjektgruppe muligens ikke har tatt i betraktning. Det åpner også for en diskusjon rundt løsningsmetodene som har blitt benyttet og vil kunne gi verdifull informasjon for gruppen vedrørende forbedringsområder.

Intervjuet vil avdekke punkter som den løpende evalueringen underveis, beskrevet i punkt 5.1.2, ikke har kunnet avdekke.

5.1.2 Løpende evaluering underveis

Ettersom vi benyttet oss av en iterativ utviklingsmetode, Scrum, var det lett å få nyttige tilbakemeldinger underveis gjennom hele prosjektet.

Produktet ble testet og evaluert etter hver iterasjon, både av prosjektgruppen og av oppdragsgiver. På denne måten ble potensielle problemer raskt korrigert og innvirkningen av nødvendige endringer minimert. Produktet ble også følgelig formet etter oppdragsgiver sine ønsker underveis.

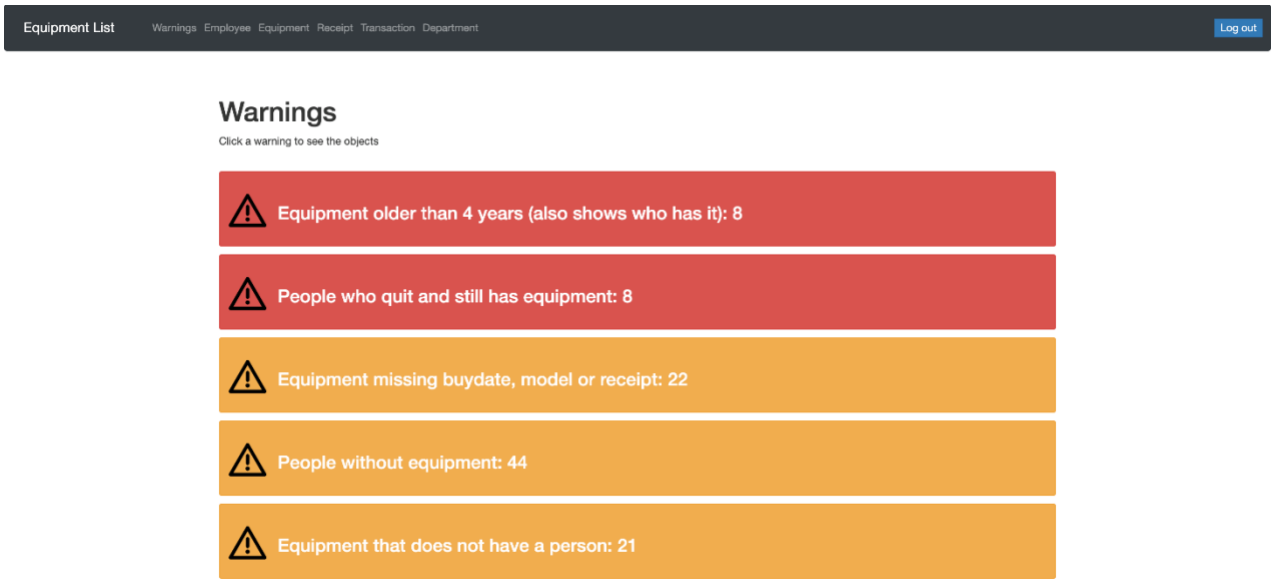
Det å kunne evaluere produktet underveis på en slik måte viste seg å være svært nyttig, da nødvendige tiltak ble gjort på et så tidlig stadium som mulig. Det var så klart en stor fordel at mesteparten av arbeidet ble gjort i SGG sine lokaler, hvor oppdragsgiver enkelt kunne oppdateres og kontaktes på kort varsel. Tiden mellom en iterasjon var ferdig og klarsignal ble gitt fra bedriften sin side for å kunne fortsette arbeidet, ble dermed minimert og mye tid ble spart på denne løsningen.

5.2 Evalueringsresultat

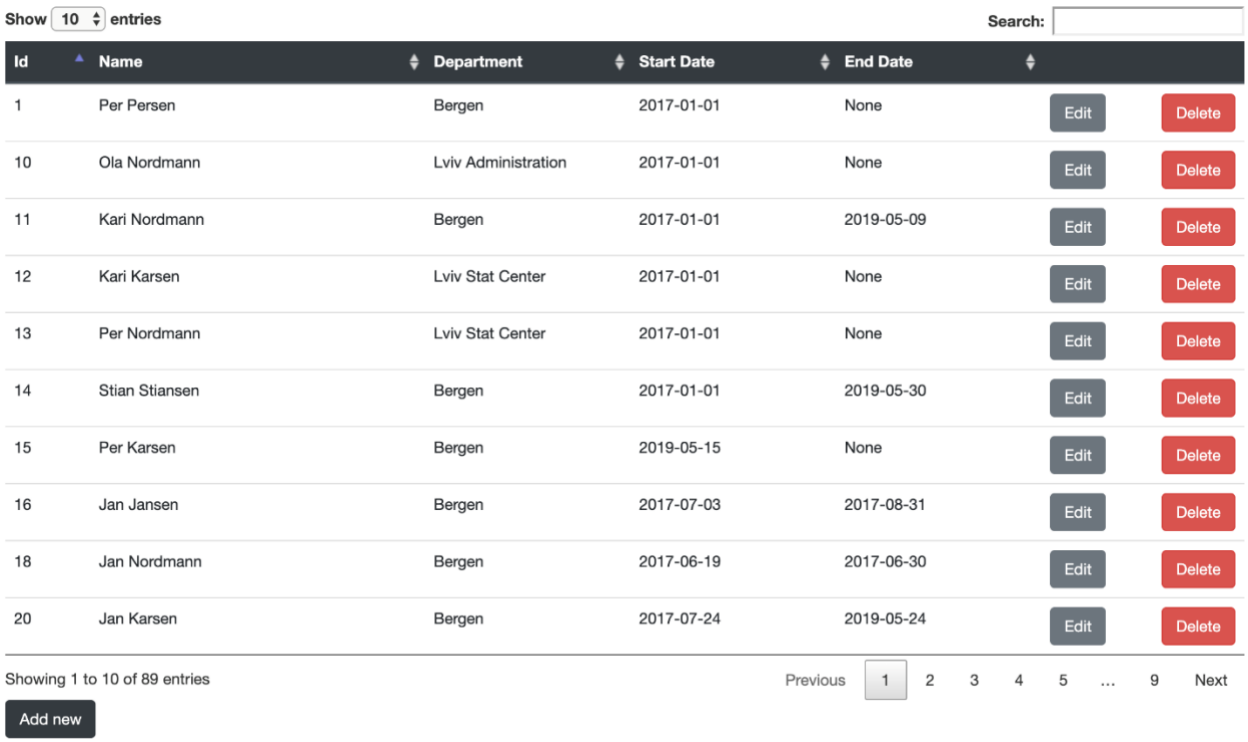
I dette punktet vil vi gå gjennom resultatet av den løpende evalueringen som ble gjort underveis gjennom hele prosjektperioden. Det er også relevant å trekke inn utdrag fra den endelige løsningen her, da den løpende evalueringen har formet sluttresultatet i stor grad. Dataene som er lagt inn her er fiktiv av hensyn til oppdragsgiver og personvern.

Eventuelle endringer som er blitt gjort etter intervjuet med oppdragsgiver vil bli beskrevet i punkt 6.

Figur 5.1 viser startsiden til applikasjonen med diverse advarsler og med en meny på toppen. Menyen er tilgjengelig fra alle sidene, noe som gjør det enkelt å bytte mellom de ulike oversiktene. Opprinnelig var det tenkt at de ulike oversiktene, som vist i figur 5.2, skulle være samlet på én side og at advarslene kun skulle vises som en liten komponent på denne siden. Det ble derimot raskt konkludert med at det ville være mer oversiktlig og brukervennlig om oversiktene var fordelt på ulike sider.

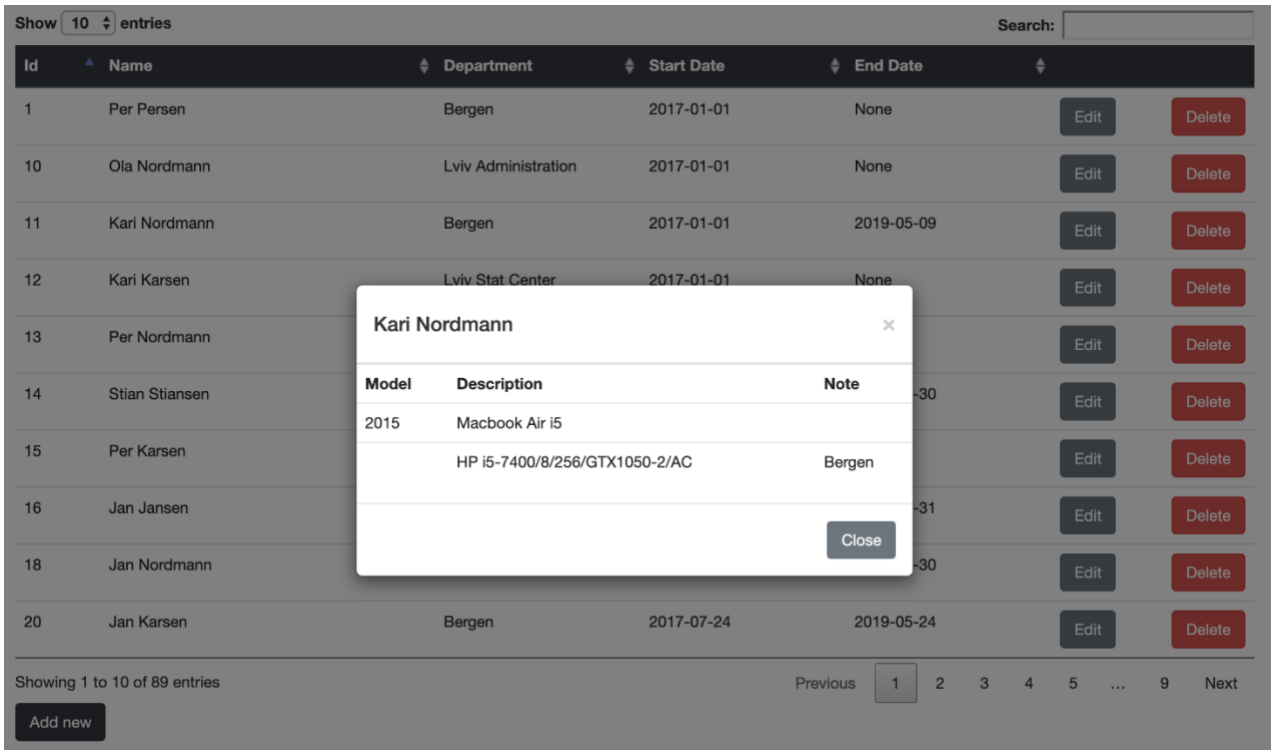


Figur 5.1: startside med opplisting av advarsler



Figur 5.2: liste over ansatte

Dersom man trykker på en av de ansatte i listen vil man få opp et vindu som viser alt utstyret som er registrert på den personen.

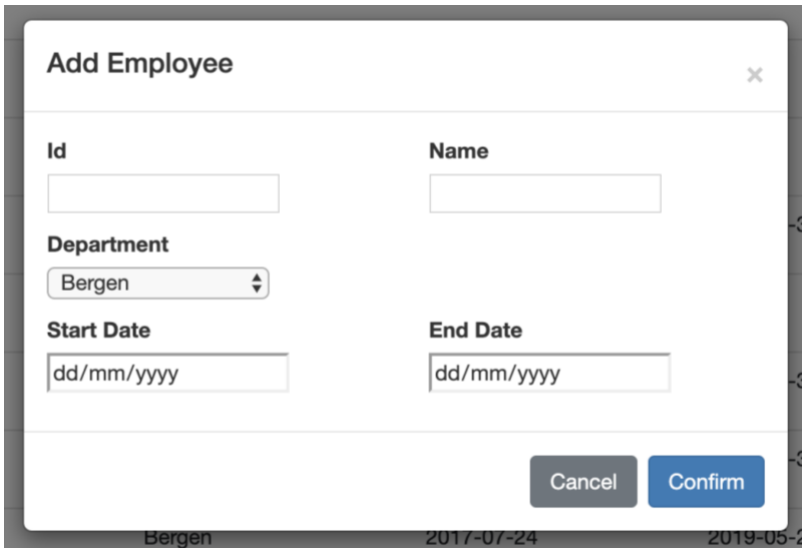


Figur 5.3: oversikt over utstyr registrert på en ansatt

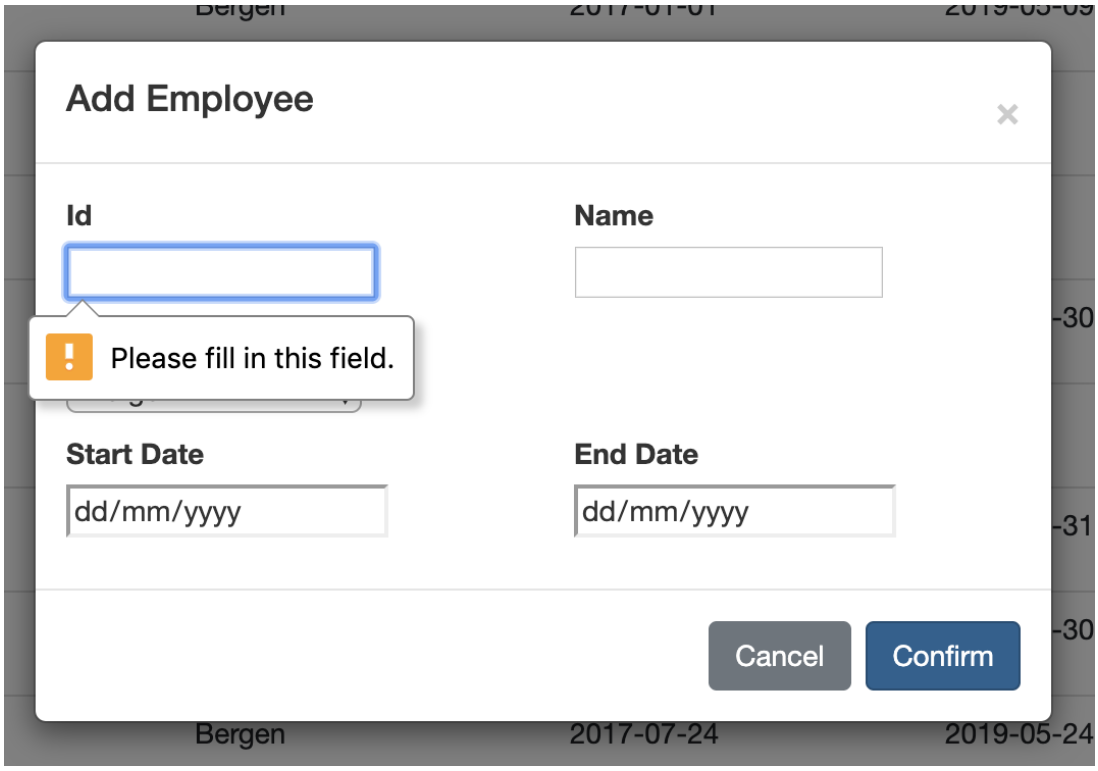
De andre oversiktssidene er lagt opp på samme måte med mulighet for sortering, filtrering, endring og sletting av felter, samt legge til nye.

Dersom en ønsker å legge til en ny rad har alle oversiktssidene en tilhørende «Add new»-knapp. Her vil det poppe opp et vindu med de nødvendige feltene for å legge til en ny enhet i databasen. Noen av feltene har også en dropdown-meny for enklere utfylling.

I den opprinnelige løsningen la denne funksjonen til en ny rad nederst i tabellen som var klar for utfylling. Etter å ha evaluert løsningen ble det klart at det ville være bedre, både visuelt og praktisk sett, om det heller ble opprettet et popup-vindu hvor man kunne legge inn all nødvendig informasjon.



Figur 5.4: skjema for å legge til en ny ansatt i databasen



Figur 5.5: varsling ved manglende data i påbudt felt

6 Resultater

Som nevnt i punkt *5.2 Evalueringsresultat* ble evalueringer gjort løpende underveis gjennom hele prosjektet. Disse evalueringsresultatene førte til at vi hele tiden jobbet i riktig retning og fikk avdekket tidlig om noe skulle utelukkes fra produktet. Dersom vi ikke hadde hatt en slik løpende evaluering underveis i prosjektet, ville vi muligens ha brukt mye tid på å utvikle et produkt i helt feil retning. Dermed ville det endelige resultatet blitt svært mangelfullt på grunn av for kort disponibel tid.

Disse løpende evalueringene viste også at oppdragsgiver var fornøyd med fremgangen i prosjektet, samt en entusiasme over å kunne ta det endelige produktet i bruk. Denne entusiasmen motiverte gruppen enda mer til å levere et så godt resultat som mulig og til å få nettsidene til å se ekstra fine ut utover det rent funksjonelle.

Vi fikk også sjekket at alle grunnleggende funksjoner var på plass, og at advarslene som ble vist på forsiden hadde rett antall knyttet til seg. Advarslene er vist i figur 5.1, og skal ved klikk ta deg til en liste med likt antall enheter som opplistet i seg.

Målet vårt var å, ved prosjektets slutt, kunne levere et komplett produkt som overholdt de kravspesifikasjoner og ønsker som oppdragsgiver hadde satt. Det overordnede målet var, som nevnt i punkt *1.1 Mål og motivasjon*, å lage et system for SGG som holder oversikt over alt utstyr som blir kjøpt inn i bedriften. Hensikten var å kunne forbedre hverdagen til de ansatte som jobber i bedriften ved å effektivisere og systematisere administrasjonen av bedriftsressursene.

Dette vil først og fremst påvirke de ansatte som regelmessig håndterer og tar seg av denne arbeidsoppgaven.

Intervjuet med oppdragsgiver ble avholdt den 15. mai, rundt en uke etter overleveringen av produktet. Transkripsjonen fra intervjuet er lagt ved under punkt 10.4. Gjennom dette intervjuet ble det tydeliggjort at Stian var fornøyd med resultatet og hvordan vi hadde valgt å løse oppgaven. Han var spesielt begeistret over søkefunksjonen vi hadde lagt inn og hvordan det var mulig å gjøre flere ting på én og samme plass. Det er blant annet mulig å liste opp utstyr som en ansatt har ved å trykke på navnet deres i listen over ansatte. En kan også finne ut hvilket utstyr en ansatt har ved å gå til listen over utstyr og søke opp navnet til personen.

Videre planlegger de å koble det opp mot G Suite slik at de ansatte med konto registrert der kan logge seg inn på området i intranettet hvor utstyrslisten vil bli tilgjengeliggjort. G Suite er en samling av skybaserte applikasjoner levert av Google. Applikasjonene som G Suite tilbyr er tilgjengelig for alle, men som bedriftskunde får man tilgang til flere innstillinger, verktøy og tjenester (G Suite, u.å.).

Bedriften har kontor i Ukraina i tillegg til i Norge, og det er derfor viktig at andre har tilgang til utstyrslisten og kan registrere nytt utstyr og tildele utstyr ved en eventuell nyansettelse for eksempel. De ønsker også å integrere det mot andre systemer og APIer slik at programmet blir så selvdrevet som mulig. For eksempel kunne det vært mulig å få listen over ansatte til å bruke APIen til regnskapssystemet og dermed unngå det ekstra arbeidet med å måtte registrere de manuelt i utstyrslistens database. Bedriften hadde derimot ingen salgsplaner for produktet og det vil derfor kun være for internt bruk i bedriften.
Selve utformingen synes han var oversiktlig og mye mer brukervennlig enn det tidligere systemet som var basert på et Excel-ark. Han var svært fornøyd med de ulike listene som var satt opp, fleksibiliteten i programmet og muligheten til å raskt og enkelt få oversikt over alt han måtte ønske.

Programmet var heller ikke overflødig og hadde kun funksjonalitet og innhold som skulle brukes, med unntak av en advarsel som kunne utelukkes fra produktet. Dersom et utstyr manglet registrert kjøpsdato, modell eller kvittering var det ikke så farlig, og det var også andre måter å kunne sjekke dette på uten å trenge å få en advarsel om det. Dette var en av de tingene oppdragsgiver i utgangspunktet ønsket og hadde satt som en del av kravspesifikasjonene, men som han etter hvert så var unødvendig.
Samtidig var det en del ting som hadde vært kjekt å ha med i forbindelse med brukervennligheten. Stian påpekte et par forbedringsområder for å gjøre det mer oversiktlig allerede samme dag som den første overleveringen av produktet. Disse områdene ble utbedret kort tid etter og en ny versjon ble presentert for oppdragsgiver før intervjuet fant sted.

Under intervjuet ble et par andre punkter nevnt, blant annet å gjøre slik at enkelte felter var forhåndsutfylt med den mest sannsynlige dataen. For eksempel vil en transaksjon i de fleste tilfeller registreres samme dag som et utstyr bytter eier, og det vil derfor være gunstig å la datofeltet være forhåndsutfylt med dagens dato. Det var også ønskelig at lenker som ble trykket på, i forbindelse med innskannede kvitteringer i Google Drive, skulle åpnes i en ny fane og ikke i samme fane.

7 Diskusjon

Dette kapittelet vil gi innsikt i de valgte tilnærmingene i prosjektet og hvilke konsekvenser de har hatt, samt potensielle forbedringer som kunne blitt gjort ved en omstart av prosjektet.

Løsningen som ble valgt for prosjektet var en webapplikasjon. Av utviklingsmetode falt valget naturlig på en iterativ metode på grunn av prosjektets lengde og prosjektgruppens størrelse.

7.1 Konsekvenser av valgte tilnærminger

Som nevnt i punkt *3.4.1 utviklingsmetode* ble det benyttet en iterativ utviklingsmetode ved navn Scrum. På grunn av den korte prosjektperioden fant vi ut at dette ville være den beste løsningen, da det åpnet og tilrettela for å kunne gjøre tilpasninger og forbedringer underveis gjennom hele prosjektperioden.

Et hovedpoeng for denne utviklingsmetoden er også at man har selvstyrte team, noe som ble gjort til en viss grad. Ettersom prosjektgruppen besto av tre personer var det begrensninger for hvordan gruppen kunne fordeles i mindre, selvstyrte team. Det ble derimot laget gode rutiner på fordeling av oppgaver, og hvert enkelt gruppemedlem tok selv ansvar for og initiativ til eget arbeid. Prosjektlederen var på den måten ikke ansvarlig for å fysisk fordele oppgaver, men heller å tilse at prosjektmedlemmene hadde noe å arbeide med til enhver tid.

Prosjektlederen var også ansvarlig for å tilse at arbeid ble gjort innen de satte fristene. Ettersom teamet samarbeidet godt, og de individuelle gruppemedlemmene i mer eller mindre grad var selvstyrte, var dette aldri noe problem. De obligatoriske innleveringene fra HVL ble gjort og levert i god tid og prosjektplanen ble fulgt opp underveis.

Enkelte deliterasjoner tok lenger tid enn planlagt, men tiden ble hentet inn igjen da andre deliterasjoner gjerne tok kortere tid enn planlagt. Det var dermed aldri noe særlig risiko for at prosjektet ikke ville bli ferdig i tide eller at produktet ikke ville bli ferdig til overleveringsdato.

Når det kommer til tilnærmingen for å løse selve oppgaven ble det tidlig avgjort at vi skulle lage en webapplikasjon. Det var en av de mer tidkrevende løsningene, men det var den løsningen som oppfylte alle de kravspesifikasjonene som oppdragsgiver hadde satt. Dette førte til at vi, nær prosjektets slutt, kunne overlevere et produkt som bedriften faktisk ville benytte seg av og ta i bruk i hverdagen. Dermed ville også et av de overordnede målene til prosjektgruppen, nemlig å forenkle hverdagen til de ansatte, oppnås.

Dersom vi hadde gått for en annen variant, for eksempel en forbedret versjon av det allerede eksisterende Excel-skjemaet, som ble diskutert under punkt *3.1 Mulige fremgangsmåter*, ville muligens ikke løsningen vår blitt tatt i bruk etter prosjektperioden. Konsekvensen av at vi valgte å gå for en webapplikasjon var derfor en høyere sannsynlighet for å oppnå de overordnede målene til gruppen.

Det at vi valgte å ta i bruk rammeverk som gruppen på forhånd ikke hadde kjennskap til, førte også til at vi var nødt til å sette av tid i starten av prosjektet for å lære oss dette. Flask er heldigvis et enkelt og svært brukervennlig rammeverk, så det påvirket ikke prosjektet i særlig stor grad tidsmessig. Fordelene med at vi valgte å gå for dette var at produktet vårt enklere kunne integreres i intranettet til SGG. Bedriften har også i stor grad benyttet seg av Python som programmeringsspråk tidligere, noe som forenklet vedlikehold og en eventuell videreføring for dem.

Kontakten mellom prosjektgruppen og både intern veileder hos bedriften og veileder på HVL har vært god. Vi valgte her en løsning hvor vi avtalte møter etterhvert som det ble nødvendig. Denne løsningen gjorde at vi alltid hadde noe konkret å ta opp på møtene og unngikk å ha møter bare for å ha dem.

7.2 Potensielle forbedringer ved omstart av prosjektet

Dersom vi skulle startet på nytt igjen med prosjektet, ville vi trolig valgt en lik tilnærming. Produktet ville blitt en webapplikasjon på lik linje som dagens resultat, men veien dit kunne vært forbedret.

Blant annet ble det brukt en del tid i starten på å sette opp tabeller med fastsatt bredde, der vi heller burde laget en dynamisk løsning fra starten av. Dette ble ordnet etter hvert og var i seg selv ikke en hindring, men mye tid kunne vært spart.

Andre løsninger vi kunne spart en del tid på er valget av programmeringsspråk. Dersom vi hadde benyttet oss av for eksempel Java til backend og rammeverk som Vue, AngularJS og React i JavaScript til frontend, ville vi ha spart tid i starten. Vi hadde satt av en del tid helt i starten av prosjektet til å sette oss inn i nye rammeverk og verktøy som skulle benyttes gjennom denne prosjektperioden. Grunnen til at vi valgte å gå for noe vi hadde mindre kjennskap til, var oppsettet av bedriftens intranett og kunnskapen som de ansatte i SGG satt med. Det ville være mer gunstig og mer praktisk for bedriften om vi benyttet oss av Python som programmeringsspråk.

Samtidig var disse prosessene svært nyttige, da de ga oss innsikt i hva som fungerte og ikke fungerte. Vi lærte mye underveis og tok med oss den kompetansen og erfaringen videre i utviklingsløpet.

Utviklingsmetoden vi tok i bruk, Scrum, synes vi derimot fungerte bra og kunne gjerne benyttet oss av samme metode ved en eventuell omstart. Planleggingen som ble gjort i forkant av prosjektet, ved hjelp av et GANTT-diagram, fungerte også bra.

8 Konklusjoner og videre arbeid

SGG er en bedrift som stadig vokser seg større, antall ansatte og utstyr registrert i bedriften øker. Det å skulle holde oversikt over ansatte, kvitteringer, utstyr, hvem som har hva og hvilken dato utstyr ble kjøpt inn, ved hjelp av et Excel-ark, var ikke lenger overkommelig da prosjektgruppen ble gitt oppgaven.

Oppgaven var svært åpen men samtidig godt spesifisert, noe som førte til at det var enkelt og raskt å komme i gang med planleggingen. Oppgaven var åpen i den forstand at det var mulig å forme produktet slik vi selv ønsket, men godt spesifisert med tanke på hva som skulle være med av funksjonalitet. Oppdragsgiver hadde kommet med en liste over konkrete funksjoner han trengte i det ferdige produktet, og prosjektgruppen kunne dermed estimere på forhånd hvor lang tid det ville ta å få inn nødvendig funksjonalitet.

Tidsestimatene ble gjort etter en vurdering av hvorvidt gruppen hadde de nødvendige forkunnskapene for å løse oppgaven eller om det ville bli nødvendig å sette av ekstra tid på dette. Resultatet ble en svært realistisk estimert tidsplan som gjenspeilet den faktiske arbeidsmengden godt. Dermed unngikk gruppen å plutselig sitte igjen med en uoverkommelig arbeidsmengde og arbeidsoppgaver som hopet seg opp.

Gruppen valgte å utvikle en webapplikasjon ved bruk av Python, PostgreSQL, HTML og litt CSS. Mye av grunnen til at valget falt på disse språkene og verktøyene var ønsker og kravspesifikasjoner fra oppdragsgiver.

Resultatet av arbeidet ble en funksjonell webapplikasjon med mulighet for å gjøre alle de ønskede handlingene knyttet til administrering av ressurser i bedriften.

8.1 Måloppnåelse

Målet med oppgaven var å lage et system for SGG som muliggjorde en effektiv og oversiktlig lagring og administrering av bedriftsressurser. Overordnet skulle dette systemet være med på å forbedre hverdagen til de ansatte i bedriften gjennom å effektivisere og forenkle et ellers tungvint arbeid.

Prosjektgruppen er selv fornøyd med arbeidet som har blitt gjort og mener det endelige resultatet er funksjonelt og har svært høy nytteverdi. Intervjuet som ble gjort med oppdragsgiver i midten av mai viste at også bedriften var fornøyd med løsningen og at den oppfylte alle kravspesifikasjonene som Stian hadde satt. De hadde allerede startet med å ta programmet i bruk under denne perioden, og kunne bekrefte at de ville bruke det videre. Etter hvert skulle det være mulig for personer fra de andre kontorene å bruke dette programmet også, noe som ville forenkle registreringen av nytt utstyr, samt registrering av tildeling av utstyr til ansatte, betydelig. Oppgaven har derfor vært en suksess og målene som ble satt i forkant av prosjektperioden er nådd.

8.2 Videre arbeid

Ved en eventuell videreføring av prosjektet ville det vært mulig å implementere følgende ønskelige funksjoner:

- Mulighet for elektronisk aksept på ansvar for utstyr
- Mulighet for å tildele ansatte et årlig budsjett på utstyr og se forbruk i forhold til budsjettet
- Integrasjon med SGG sin intranettside og database med blant annet felles innlogging og ansattregister
- Integrasjon til ERP-systemet Xledger via API for å hente ut data om ansatte og nye bilag på utstyr
- Mulighet for å legge inn forespørsler fra ansatte og kommentarer fra ledere på ønsket utstyr
- Funksjoner som kan forbedre og forenkle prosessen med innkjøp
- Full sporbarhet på hvilken bruker som har gjort endringer i databasen

Overnevnte punkter er ønskelige funksjoner som oppdragsgiver informerte om fra starten av, men som på grunn av prosjektets lengde ikke var mulig å få implementert.

Det kunne også vært mulig med tilgjengeliggjøring av produktet for andre bedrifter. Det har ikke blitt gjort noen undersøkelser på etterspørselen etter en slik webapplikasjon, så vi kan dermed ikke konstatere om dette hadde vært aktuelt eller ikke. Omfanget av en slik videreføring hadde vært svært stort da den nåværende løsningen er tilpasset til SGG sitt bruk med en del statiske løsninger.

En siste ting som hadde vært lur å implementere er en funksjon som regner ut garantitid etter hvilken type utstyr det er. Advarselen som er lagt inn nå benytter seg av informasjonen om hvor lenge utstyret har vært i bruk for denne kalkulasjonen.

9 Litteratur/referanser

1. API (u.å.), i: *SNL* [Internett]. Tilgjengelig fra: https://snl.no/API [Lest 23. april 2019]
2. Bootstrap (u.å.) Bootstrap [Internett]. Tilgjengelig fra: https://getbootstrap.com/ [Lest 28. mai 2019]
3. Design Patterns - MVC Pattern (u. å.) [Internett]. Tilgjengelig fra: https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm [Lest 29. mai 2019]
4. Enterprise Resource Planning (u.å.), i: *Wikipedia* [Internett]. Tilgjengelig fra: https://no.wikipedia.org/wiki/Enterprise_resource_planning [Lest 14. april 2019]
5. Flask (u.å.) Welcome | Flask (A Python Microframework. [Internett]. Tilgjengelig fra: http://flask.pocoo.org/. [Lest 12. mars 2019].
6. G Suite (u.å.) G Suite: bedriftsapper for samarbeid og produktivitet. Tilgjengelig fra: https://gsuite.google.no/intl/no/ [Lest 28. mai 2019].
7. Nodejs (u.å.) About Node.js [Internett]. Tilgjengelig fra: https://nodejs.org/en/about/ [Lest 12. mars 2019]
8. Oracle (u.å.) Java Servlet Technology [Internett]. Tilgjengelig fra: https://www.oracle.com/technetwork/java/servlet-138661.html [Lest 12. mars 2019]
9. Scrum (u.å.) What is Scrum? [Internett]. Tilgjengelig fra: https://www.atlassian.com/agile/scrum [Lest 9. april 2019]
10. SQLAlchemy (u.å.) SQLAlchemy – The Database Toolkit for Python [Internett]. Tilgjengelig fra: https://www.sqlalchemy.org/. [Lest 12. mars 2019].

10 APPENDIX

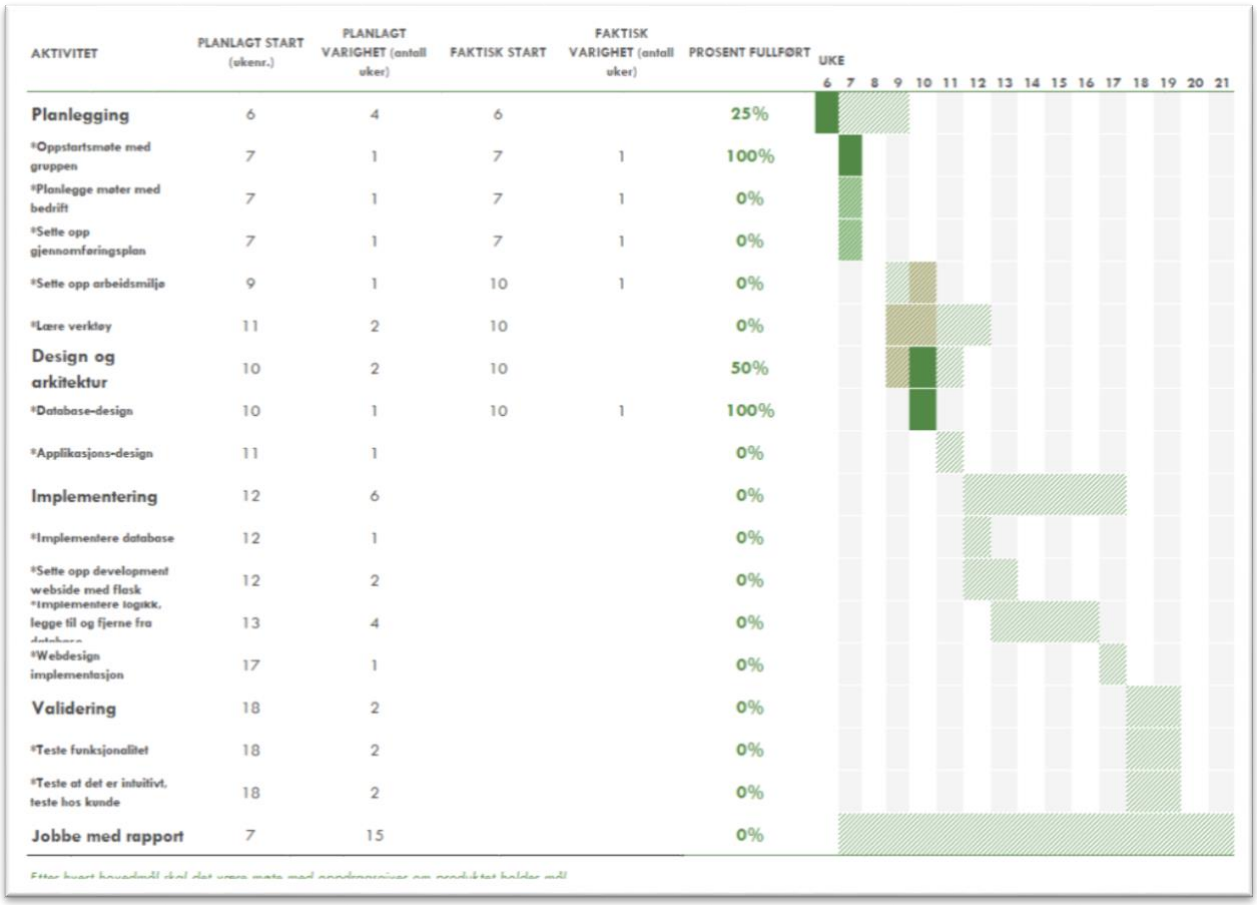
10.1 Risikoliste

Risiko	Sannsynlighet	Alvorlighetsgrad	Tiltak
Sykdom	Middels	Middels	Passe på at ingen av gruppemedlemmene blir overarbeidet, da dette kan øke risikoen for sykdom. God kommunikasjon mellom medlemmer for å tilse at arbeid blir gjort.
Dårlig samarbeid innad i gruppen	Lav	Høy	Bli enig om regler på forhånd. Sørge for at det er god kommunikasjon mellom gruppemedlemmene. Avholde møte hvor vi kommer frem til en felles løsning.
Dårlig kommunikasjon med oppdragsgiver	Lav	Middels	Tilby jevnlige oppdateringer på hva vi jobber med og har i tankene.
Skjevfordeling av arbeid	Lav	Høy	Fordele arbeid etter evner og sørge for at alle gruppemedlemmer har oppgaver de klarer å løse.
Manglende kompetanse	Middels	Middels	Sette av tid til å sette oss inn i nye verktøy, programmeringsspråk og rammeverk.
Misforståelser knyttet til kravspesifikasjonene	Lav	Høy	Stille spørsmål dersom noe er uklart. Kravspesifikasjonene er godt dokumentert, men misforståelser kan allikevel oppstå.
Utstyr som ikke fungerer	Middels	Middels	Ha tilgjengelig alternativt utstyr. Alltid sikkerhetskopiere og sørge for at prosjektet er tilgjengelig fra GitHub.
Dårlig tidsplanlegging	Middels	Middels	Sørge for at vi holder tidsplanen vi har satt og justere kursen underveis om nødvendig.
Når ikke satte mål	Middels	Høy	Planlegge realistisk. Jobbe videre og prøve å nå neste mål.

Tabell 1: risikoliste

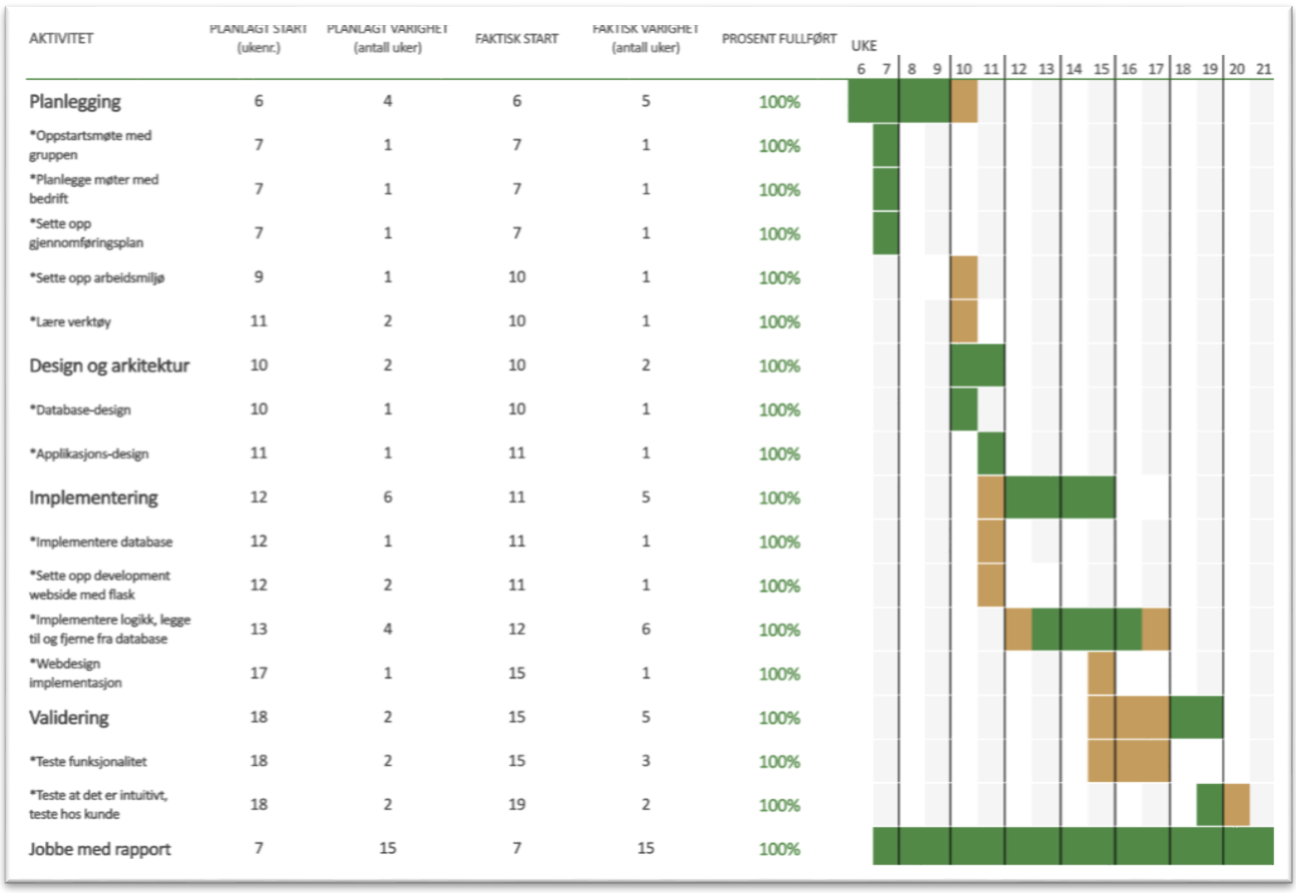
10.2 GANTT diagram

10.2.1 Opprinnelig GANTT-diagram:



AKTIVITET	PLANLAGT START (ukenr.)	PLANLAGT VARIGHET (antall uker)	FAKTISK START	FAKTISK VARIGHET (antall uker)	PROSENT FULLFØRT
Planlegging	6	4	6		25%
*Oppstartsmøte med gruppen	7	1	7	1	100%
*Planlegge møter med bedrift	7	1	7	1	0%
*Sette opp gjennomføringsplan	7	1	7	1	0%
*Sette opp arbeidsmiljø	9	1	10	1	0%
*Lære verktøy	11	2	10		0%
Design og arkitektur	10	2	10		50%
*Database-design	10	1	10	1	100%
*Applikasjons-design	11	1			0%
Implementering	12	6			0%
*Implementere database	12	1			0%
*Sette opp development webside med flask	12	2			0%
*Implementere logikk, legge til og fjerne fra	13	4			0%
*Webdesign implementasjon	17	1			0%
Validering	18	2			0%
*Teste funksjonalitet	18	2			0%
*Teste at det er intuitivt. teste hos kunde	18	2			0%
Jobbe med rapport	7	15			0%

10.2.2 GANTT-diagram ved prosjektets slutt:



AKTIVITET	PLANLAGT START (ukenr.)	PLANLAGT VARIGHET (antall uker)	FAKTISK START	FAKTISK VARIGHET (antall uker)	PROSENT FULLFØRT
Planlegging	6	4	6	5	100%
*Oppstartsmøte med gruppen	7	1	7	1	100%
*Planlegge møter med bedrift	7	1	7	1	100%
*Sette opp gjennomføringsplan	7	1	7	1	100%
*Sette opp arbeidsmiljø	9	1	10	1	100%
*Lære verktøy	11	2	10	1	100%
Design og arkitektur	10	2	10	2	100%
*Database-design	10	1	10	1	100%
*Applikasjons-design	11	1	11	1	100%
Implementering	12	6	11	5	100%
*Implementere database	12	1	11	1	100%
*Sette opp development webside med flask	12	2	11	1	100%
*Implementere logikk, legge til og fjerne fra database	13	4	12	6	100%
*Webdesign implementasjon	17	1	15	1	100%
Validering	18	2	15	5	100%
*Teste funksjonalitet	18	2	15	3	100%
*Teste at det er intuitivt, teste hos kunde	18	2	19	2	100%
Jobbe med rapport	7	15	7	15	100%

10.3 Akronymer

SGG	Scout Gaming Group
HVL	Høgskulen på Vestlandet
SQL	Structured Query Language
ERP	Enterprise Resource Planning
API	Application Programming Interface
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
MVC	Model View Controller

Tabell 2: akronymer

10.4 Transkripsjon fra intervju

S: Stian. PG: Prosjektgruppen.

PG Du må bare bekrefte at det går fint at vi tar opptak av dette og at det går fint å bruke

S Ja

PG Så nå har du jo fått testet ut programmet litt så det første vi lurer på da er om du føler at det oppfyller alle kravspesifikasjoner som er satt.

S Ehm.. oi nå husker jeg ikke kravspesifikasjonene. Skal vi bare gå gjennom de?

PG Ja, vi kan gjerne gå gjennom de.

S Ja, har dere de foran deg? Eller så kan jeg sikkert finne de.

PG Her.

S Registrere nytt utstyr og slette utstyr som er registrert feil. Eh, ja, absolutt. Laste opp kvittering og lenke til kvittering i Google Drive fungerer veldig bra. Registrere hvem som har fått hvilket utstyr.. det holder jeg på med nå. Så det fungerer bra. Få oversikt over hvilket utstyr en ansatt har. Det har jeg sett.. nå har jeg ikke lagt inn så mange her, men hvis jeg nå tester en her.. employee.. Tove.. der står det at hun har to MacBook Air og to Lenovo.. så der har det skjedd en dobbeltregistrering. Men jeg får oversikt. Hege.. en MacBook Air. Veldig bra. Det gjør jeg. Få oversikt på hvilket utstyr som er ledig og hvor det befinner seg. Ja, det er vel bare til å gå på utstyrslisten og så ser jeg hva som står på «none» på Employee.. så de er hvertfall ikke registrert på noen ansatte.

PG Så er det og en «Warning» på det.

S Ja, det er det og ja. Skal vi se.. hvis jeg søker på «none» hvis jeg søker på utstyrslisten så går det vel på selve utstyret og ikke employee, men uansett så..
«Equipment that does not have a person», det har jeg skrevet opp på listen. Som ikke har en person. Heh, en litt merkelig måte å skrive det på men der har vi alt utstyret som jeg ikke har registrert enda ja. 103 stykk.. så den er jo glimmers. Ehm, det hender at en ekstra har en.. at en ansatt har en ekstra laptop som de ikke trenger. Utstyr må derfor kunne registreres som ledig selv om det er tildelt en ansatt. Har vi implementert det? Jeg tror ikke vi gjorde det.

PG Nei, tror ikke det. Tror vi snakket om det.

S Men det er egentlig ikke så veldig viktig behov. Når jeg tenker meg om. For da står det ofte som.. vi kan bare ha det som «none», altså vi trenger ikke tildele det noen. Men jeg kan notere på utstyret at..

PG Det vil jo stå i transaction hvem som har det.

S Ja, men da ser jeg ikke at det er ledig selv om det står på en person. For da må jeg sette det tilbake til «none». Men det jeg kan.. ja.. men jeg kan bare notere det på utstyret at denne her har en person.. en person har den foreløpig men den skal ut. Så jeg tror det ikke er så viktig når jeg tenker meg om. Få oversikt på hvem som har hatt et utstyr som er blitt kjøpt. Ja, det er bare til å sortere i transactionlisten. Jeg liker det at du kan bruke forskjellige lister til forskjellige ting, eller til mange forskjellige ting. At du ikke bare har ett sted i menyen som gjør akkurat den spesifikke tingen.. du har liksom.. dermed kan du bruke det til ting du ikke har tenkt på engang. Så hvis jeg tar id 11 her, så ser jeg at den Asusen har vært Øystein sin og nå er den «gone» og det er fordi den ble ødelagt.

Veldig bra. Få varsel om utstyr som mangler kvittering, bilag eller lokasjon. Om det mangler bilag går jo og an å bare se i utstyrslisten selv om det sikkert er en warning på det, er det ikke det?

PG Jo.

S Equipment.. people without equipment.. eller det er det forsåvidt ikke. Older equipment..

PG Men hvis de ikke har noe så blir de fjernet. Hvis den listen var tom så skulle den jo ikke vises, men hvis du ser i min da.. equipment missing buy date or receipt. Han vises ikke dersom han er null.

S Okei, men vi har jo utstyr som ikke har receipt.

PG Da skal vi ta å bugfixe det.

S Den har vel tatt en eller istedet for en og eller en og i stedet for en eller.. vet ikke.. er du med?

PG Jeg er helt med. Vi kan fikse det.

S Men det er forsåvidt ikke så viktig heller. Altså.. fordi, jeg har jo funnet ut at greit, det er noen ting som hadde vært kjekt som jeg ikke har tenkt på men noen ting som jeg trodde hadde vært kjekt som egentlig ikke er viktig. Fordi, jeg trenger for så vidt ikke advarsel, det er ikke farlig om jeg ikke har registrert en kvittering på noe. Fordi at jeg har andre rutiner jeg kan ta og så har jeg i utstyrslisten her så kan jeg bare sortere etter receipt og hvis det står none så er det ikke registrert kvittering. Det er jo superenkelt. Så den har dere forsåvidt implementert selv om det var en liten bug der. Men.. trenger det egentlig ikke. Ser ut som.. kvittering, bilag, lokasjon.. ja, lokasjon har vi jo. Det er jo advarsel om at den ikke er tildelt noen. Kvittering og bilag er det samme. Få varsel om hvilket utstyr som er klart for utskiftning. Den har jeg sett. Og så er det noen ønskelige funksjoner, ja det er vel neste spørsmål kanskje. Så.. kravspesifikasjonen, check, veldig bra.

PG Den er i orden, ja. Men er det noe av det vi har laget nå som du tenker ikke har særlig nytteverdi.. som vi bare kunne fjernet fra programmet egentlig?

S Nei, ehm, som dere har laget ekstra eller som jeg har bedt om som vi egentlig ikke trenger?

PG Begge deler.

S Ja, den advarselen som du fikk opp der, den kan vi egentlig bare ta vekk. Om at det mangler bilag. Den trenger jeg ikke. Og.. mangler lokasjon. Det er den advarselen om equipment that does not have a person.. hehe.

PG Jeg kan skrive om den formuleringen, hehe.

S Jeg har skrevet den på en liste.. absolutt ikke viktig, men hvis dere har lyst til å fikse småting, så er det den.. jeg har noen andre artige også, skal vi se.. receipt, add new, så har du supplement har dere kallet første feltet der, og det er vel bilagsnummer.

PG Ja, vi visste ikke hva det var på engelsk så vi trodde det var det vi..

S Ja, bilagsnummer på engelsk.. kall den receipt number eller voucher number. Ikke viktig. Overhodet ikke. Skjønner absolutt hva vi har skrevet. Ellers så tror jeg den er ganske, altså jeg kommer jo på ting i ettertid som hadde vært kjekt men det blir en ny oppgave. Så, det er ingenting.. dere har ikke laget noe unødvendig ting synes jeg altså. Det har vært ganske on point og dere har løst ting veldig bra. Jeg liker listene, jeg liker fleksibiliteten, mulighet til å få oversikt, jeg liker det der søkefeltet.. altså hadde jeg ikke hatt det så hadde jeg kommet med en haug med andre ting jeg trengte, men det er sånn jeg ikke visste at jeg trengte før jeg fikk det. Så det er helt genialt. Det fulgte vel med GUI-pakken dere har brukt?

PG Ja.

S Dere skal få æren av den likevel.

PG Har dere noen plan videre for dette her nå.. altså skal dere videreutvikle eller har dere.. noe plan?

S Ja, jeg har tenkt noen planer. Nå må vi jo ta det i bruk først, men.. jeg.. mens jeg bruker det så skriver jeg ned alle sånne småting. Som hadde gjort det mer effektivt og mer oversiktlig, så tenkte jeg og etterhvert at det er greit at andre også tar det i bruk sånn at når en eller annen i Ukraina.. en eller annen HR-ansvarlig eller noe sånn setter opp utstyr til en ny ansatt som begynner, så slipper jeg å gå inn å spørre hva utstyr han har fått og finne frem og registrere det. Og det bare å bruke verktøyet til å se hva utstyr vi har her nå og hva trenger vi, og registrere det på den.. bruker det gjerne i sammenheng med dette skjema vi nå har for at de får ansvar for utstyret. Ellers er det å ha det på intranettsiden.. en eller annen innlogging der man bruker katalogtjenesten til Google.. G-Suite.. sånn at de ansatte som har konto der, de får logget seg inn på dette området og bruke utstyrslisten. Og så har vi lyst til å integrere det med andre ting. For eksempel ansattlisten.. at den bruker API-en til regnskapssystemet. At den finner ansatte der. For nå er det litt.. det jeg liker å kalle «bullshit in and bullshit out»-system. Du får verdiene.. dataene du får ut er ikke bedre enn den du putter inn. Og det gir meg en verdi.. fordi det gir meg en bedre oversikt over det jeg får på et Excel-ark. Og mye mer brukervennlig og solid. Men samtidig, hvis jeg ikke skriver at en ansatt har sluttet så får jeg ikke advarsel om at den ansatte som har sluttet fremdeles har utstyr. Hvis jeg ikke registerer at de har levert fra seg utstyr, så hjelper det heller ikke med de advarslene. Så det å få sånne ting til å gå litt mer automatisk. Det blir liksom neste steg.

PG Ja, men, så det blir kun for å bruke internt i bedriften? Ikke noe.. salgsplaner for det?

S Nei, vi har ikke tenkt å tjene penger på det. Og det er.. altså.. vi ville ha det i utgangspunkt for å få.. for å slippe å bruke egne ressurser på det, for de må vi bruke på produktet vårt. Og for å teste gode elever.. og komme i kontakt med kjekke folk. Så, det var målet. Og det kan jo hende vi har lyst til å utvikle det videre og sånn, da får vi håpe at vi får litt av koden. Eller et eller annet vi kan bygge på. Eller at vi kan bruke dere og finne en deal. Men jeg er veldig entusiastisk om dere har lyst til å bruke dette til andre. Så synes jeg at det er supert. Det er deres programvare. Men jeg vil bruke det.

Jeg hadde syntes det var gøy hvis dere kunne brukt det til noe mer. Men det blir en hel businessidé.. eller jeg vet ikke hvor mye verdi det ligger i det.. men det.

PG Vanskelig å si. Det er veldig mange lignende ute.

S Ja, og så er det det å få det tilpasset til akkurat det du trenger. Jeg så innpå andre verktøy men det er slitsomt å sette seg inn i noen voldsomme greier, og så blir det enten.. enten mangler det funksjonalitet eller så blir det for voldsomt. Og så kan det og bli idiotisk dyrt. Så jeg tenkte at en såpass definerbar bacheloroppgave ikke var dumt. Kanskje jeg tar alt jeg kunne tenkt meg å forbedre og lage en ny bacheloroppgave ut av det.

PG Synes du det var enkelt nok å bruke? Altså er det oversiktlig nok, eller er det noe vi kunne forbedret for å gjøre det mer oversiktlig?

S Synes det var veldig, veldig bra oversiktlig.. veldig bra brukervennlighet. Det er klart.. jeg har funnet noen småting som ville gjort det mer effektivt. For eksempel fokus på GUI-elementer, når du trykker new så er første feltet klart til å skrives i i steden for at du må trykke der eller trykke på tab.. at du legger inn litt sånn at han setter fokus på den første tingen og når du trykker tab så hoppe den til neste hvor du skal skrive inn noe. Og kanskje sånn at noen av feltene du skal skrive inn noe er ferdig utfylt med den mest sannsynlige dataen du skal bruke, for eksempel dagens dato på transaksjoner.

Men det er mye sånne småting, det er umulig å tenke på alle sånne ting før du virkelig har begynt å bruke det skikkelig. Så jeg synes at det er veldig bra.

PG Vi har egentlig ikke noen flere spørsmål ut over det. Du må ikke være redd for å spørre oss om å fikse på småting.

10.5 Oppgavebeskrivelse

P34 - Utstyrsliste i database med webgrensesnitt (Scout Gaming)

Om oss

Scout Gaming Group (www.scoutgaminggroup.com) er et konsern som ble startet av to pokerspillere fra Bergen i 2013. I 2017 ble selskapet børsnotert i Sverige og nå er vi ca. 80 ansatte fordelt på kontorer i Bergen, Stockholm, Ukraina og Malta. Scout lager og drifter en spillplatform som brukes til såkalt «Fantasy Sports» og «pool betting»-spill. Våre kunder er andre spillselskap som ønsker å tilby Fantasy Sports til sine spillere som Betsson og f.eks. Norsk Tipping. Vi leverer også spill til mediaselskap som f.eks. VM Manager til TV2 og NRK.

Utviklingen i Scout drives fra Kanalveien 52C i Bergen hvor ca. 20 av oss jobber. Dette er bare 5 minutter gangavstand fra Campus Kronstad. Her sitter også konsernets økonomisjef, Stian Lysvold Haugse, som vil være ansvarlig for prosjektet. Stian har en bachelorgrad fra Høgskolen i programmering fra 2003 i tillegg til økonomi og administrasjon fra 2013. Helene Dunlop skal ha ansvar for teknisk bistand og veiledning. Helene er utvikler med bachelor fra Høgskolen fra 2017.

Bakgrunn

Antall ansatte i Scout Gaming har doblet seg på halvannet år. Som teknologibedrift er det viktig at utviklere får det utstyret de trenger til å utføre jobben. I 2018 kjøpte vi datautstyr for ca. 700.000 kroner så det er en ressurs som er viktig å ha kontroll på. For tiden har vi et regneark som gir oss foreløpig den kontrollen vi trenger. Men det er ikke enkelt å bruke og vi mister lett oversikten hvis noe ikke blir registrert som det skal. Den beste løsningen er å legge dette inn i en database med tilgang via vår intranettside.

Vi kunne laget dette selv, men vi må prioritere ressursene til selve produktet vårt. Å tilby denne oppgaven som et hovedprosjekt til Høgskolen vil forhåpentligvis også tiltrekke noen gode kandidater som vi kan ansette.

Oppgaven

Oppgaven består i å lage en database av utstyrslisten og hvem som har hva. Denne må kunne brukes via et webgrensesnitt på Scout sine intranettsider. Denne oppgaven kan evt. deles mellom studenter fra forskjellige studieretninger der noen tar seg av back-end programmering, noen tar websiden og noen setter opp databasen og webserver.

Krav til funksjoner

Webgrensesnittet og databasen må kunne gi oss følgende funksjoner og rapporter:

- Registrere nytt utstyr og evt. slette utstyr som er registrert feil
- Laste opp kvittering eller linke til kvittering i Google Drive
- Registrere hvem som har fått hvilket utstyr
- Få oversikt på hvilket utstyr en ansatt har
- Få oversikt på hvilket utstyr som er ledig og hvor det befinner seg
- Det hender ansatte har en ekstra laptop som de ikke trenger. Utstyr må derfor kunne registreres som ledig selv om det er tildelt en ansatt.
- Få oversikt på hvem som har hatt et utstyr siden det ble kjøpt
- Få varsel om utstyr som mangler kvittering, bilag eller lokasjon
- Få varsel om hvilket utstyr som snart er klart for utskiftning

Ønskelige funksjoner

- Følgende funksjoner er ikke et krav, men om det ikke implementeres må systemet bør være tilrettelagt så det kan være mulig å utvikle senere:
- Mulighet for elektronisk aksept på ansvar for utstyr
- Mulighet for å tildele ansatte et årlig budsjett på utstyr og se forbruk i forhold til budsjettet
- Integrasjon med vår intranettside og database med bl.a. felles innlogging og ansattregister
- Integrasjon til ERP-systemet Xledger via API for å hente ut data om ansatte og nye bilag på utstyr
- Mulighet for å legge inn forespørsler fra ansatte og kommentarer fra ledere på ønsket utstyr og evt. andre funksjoner som kan forbedre og forenkle prosessen med innkjøp
- Full sporbarhet på hvilken bruker som har gjort endringer i databasen
- Brukervennlighet er viktig i Scout, så vi blir imponert hvis oppgaven er løst med høyt fokus på UX.

Teknologi

I Scout bruker vi PostgreSQL, så det er ønskelig at databasen bygges på samme DBMS. Webgrensesnittet bygges i utgangspunktet med vanlig html og kanskje javascript. Som rammeverk stiller studentene fritt til å velge f.eks. Nuxt, Vue, Angular, React eller kanskje Scout sin egenutviklede open source Widgets (github.com/scoutgg/widgets). Våre utviklere vil være behjelpelig ved valg av teknologi.

Hvorfor velge denne oppgaven?

Et hovedprosjekt har en tendens til å kreve nesten uoverkommelig mye arbeid. Denne oppgaven er i utgangspunktet ganske enkel. Samtidig kan den skaleres til å bli nok utfordrende. Men den viktigste grunnen er at du vil komme i kontakt et fantastisk utviklermiljø som bl.a. sitter på Bergens største kompetanse på PostgreSQL. Vi krever ikke at du kan alt fra før, men hvis du viser engasjement og arbeidsvilje så kan Scout Gaming være inngangen til en god karriere.

Studenter som velger denne oppgaven får kontorplass i våre lokaler om de ønsker det. Her kan vi lokke med pizza i fryseren, god kaffemaskin, loungeavdeling med tribune, 70" TV, Playstation, bordtennis og ikke minst en bar fyllt opp med øl.

Kontaktpersoner

Stian Lysvold Haugse
Økonomisjef
SLH@scoutgg.com
tlf 90 66 99 18

Helene Konstantine Dunlop
Programutvikler
HDU@scoutgg.com
tlf 418 13 444

10.6 Arkitektur