

DAT17A

NAVIAIR

25.05.2018



MARTINE GARDE MAGH
03-05-93

VERONIQUE CACHO BASSON JENSEN
17-09-1992

Indholdsfortegnelse

Indledning.....	1
Problemstilling.....	1
Samarbejde.....	1
ITO.....	2
Beskrivelse af virksomheden	2
Forretningsgrundlag.....	3
Organisationsform	3
SWOT – analyse	4
Styrker	5
Svagheder	5
Muligheder	6
Trusler.....	6
Feasibility study	7
Tekniske forhold.....	7
Organisatoriske og operationelle forhold	7
Interessent analyse	8
Risikoanalyse	10
Glossary.....	11
Unified process	12
Inception-fasen.....	12
Systemvision	12
Kravanalyse.....	12
FURPS	15
Functionality	15
Usability	15
Reliability	15
Supportability.....	16
Use case model.....	17
Aktørbeskrivelser:	17
Domæne model	18
Elaboration-fasen.....	19
Use case model	19

System sekvensdiagram	19
Domæne model	20
Fully dressed use cases.....	21
UC1: manageEquipment.....	21
UC2: searchEquipment.....	23
Construction-fasen.....	26
Klassediagram.....	26
SD diagram.....	27
Database.....	27
Beskrivelse af tabellerne	28
Normalisering.....	29
Relations skema	30
Overgangen fra klassediagram til relations skema	32
Interessante kode beskrivelser.....	33
Vurdering af UP-metoden	36
Inception.....	36
Elaboration	36
Construction	36
Konklusion	37
Kode henvisning.....	37
Litteraturliste.....	38
Bilag 1.....	40
Bilag 2: Vejledning til Naviair Archives.....	41

Indledning

Vi har efter vores to første semestre på datamatikeruddannelsen på Københavns Erhvervs Akademi, fået til opgave at udvikle en databaseapplikation. Applikationen skal være lavet på baggrund af et behov hos en virksomhed, som vi selv har skulle finde og få et samarbejde op at køre med.

Vi har derfor fået lavet en aftale med firmaet Naviair, der tager sig af al trafikstyring i det danske luftrum. De havde brug for et databasesystem der skulle kunne holde styr på al deres IT-udstyr. Vores rapport er delt op i tre dele – informationsteknologi i organisationer, software design og software konstruktion – og vi vil igennem rapporten gøre brug af de værktøjer, kompetencer og metoder vi har lært indtil videre i vores tid på KEA.

Problemstilling

I Naviair er sikkerheden alfa og omega. De sørger hver dag for at omkring 1900 flyvninger, både dem der passere dansk luftrum, og dem der letter og lander, foregår sikkert og uden komplikationer. Derfor er det også yderst nødvendigt at alle deres systemer ikke kan blive hacket eller på anden måde destrueret af ude kommende.

En af de måder de sørger for at dette ikke sker, er f.eks. ved at holde faciliteterne adskilt i deres IT-udstyr. Det er således ikke alle deres computere der kan komme på nettet, lave flyveplaner, yde mailservice eller man kan tjekke radar på, men kun nogle enkelte.¹

Derfor ville Naviair godt have lavet et arkivsystem, der skal holde styr på alt deres IT-udstyr, hvilke apps der er installeret, og hvilken service man kan bruge udstyret til.

Vi vil i denne opgave prøve at lave et program der indeholder oplysninger om alt Naviairs IT-udstyr, og en søgefunktion, der gør det muligt at finde et bestemt IT-udstyr, der kan yde den service man har behov for.

Samarbejde

Da vi kun er to i vores gruppe, har vi valgt at udarbejde alle opgavens dele i fællesskab, i stedet for at dele den op og lave tingene individuelt. Vi vil gerne fokusere på at samarbejde og kommunikere med hinanden, sådan at vi begge to forstår det produkt vi laver. Der er derfor ikke defineret i opgaven, hvem der har lavet hvilke dele.

¹ Dette fremgik af mødet med Naviair.

ITO

Beskrivelse af virksomheden

Naviair er en leverandør af luftfartstjeneste i Danmark. Deres opgave er at holde styr på al trafik i det danske luftrum, og sørge for at lede fly sikkert og uden forsinkelser gennem luftrummet.

Naviair startede ud med at være en del af Statens Luftfartsråd, men blev i 2001 til en selvstændig statsvirksomhed, da man ville sikre adskillelse mellem luftfartstjenesten og tilsynsmyndigheden. I maj 2010 vedtog Folketinget at Naviair i stedet skulle etableres som en selvstændig offentlig virksomhed, og den er den dag i dag stadig ejet af den danske stat under Transportministeriet.²

Naviair har kontrollårne i København, Billund, Aalborg, Aarhus, Roskilde og Bornholms lufthavne.

Kontrolcentralen ligger i Kastrup lufthavn og her bliver al trafik overvåget fra.

Centralen er bemanded døgnet rundt, alle dage om året. Her håndterer de hvert år omkring 700.000 flyvninger, og iblandt disse er 60 procent flyvere, der bare passere det danske luftrum, uden at lette eller lande. Naviair leverer også flyveinformationstjeneste i det grønlandske luftrum og på Færøerne.³

Naviair er blandt de bedste leverandører i Europa når det gælder standardisering og harmonisering af lufttrafikstyringssystemer. Kontrolcentralen i København og fire andre europæiske landes kontrolcentraler (Sverige, Østrig, Kroatien og Irland) har sammen formet COOPANS Alliance.

De samarbejder om i fællesskab at opgradere, harmonisere og udvikle deres lufttrafikstyringssystemer, sådan at alle systemerne bruger fælles software og at vedligeholdelsen er harmoniseret.⁴

Naviair har på nuværende tidspunkt 711 ansatte, hertil kan det være at der er eksterne konsulenter. De har en årlig omsætning på omkring 750 millioner kroner.⁵

² (Check-in, 2010)

³ (Naviair - Det laver vi, u.d.)

⁴ (Naviair - COOPANS, u.d.)

⁵ Fremgik af mødet med Naviair

Forretningsgrundlag

Naviairs mission er at hjælpe med at skabe værdi og velfærd for samfundet. Dette skal ske ved hjælp af udvikling og levering af sikker, effektiv trafikstyring til luftfarten, til konkurrencedygtige priser. Dette gør at de udfylder deres rolle som et afgørende led i luftfartens værdikæde.

Naviairs vision for fremtiden er forblive blandt de bedste leverandører af luftfartstjenester i Europa. Deres virksomhed skal forblive i konstant udvikling, og derved sikre deres stærke position blandt kunder og samarbejdspartnere. De vil forsøge at leve branchens bedste produkter til konkurrencedygtige priser, og prioritere høj sikkerhed, service og kvalitet.

Disse mål vil de opnå gennem deres motiverede og engagerede medarbejdere, og en arbejdsplads med store krav, hvor målrettet medarbejderudvikling og involvering er grundlaget for opretholdelsen af en attraktiv virksomhed.⁶

Organisationsform

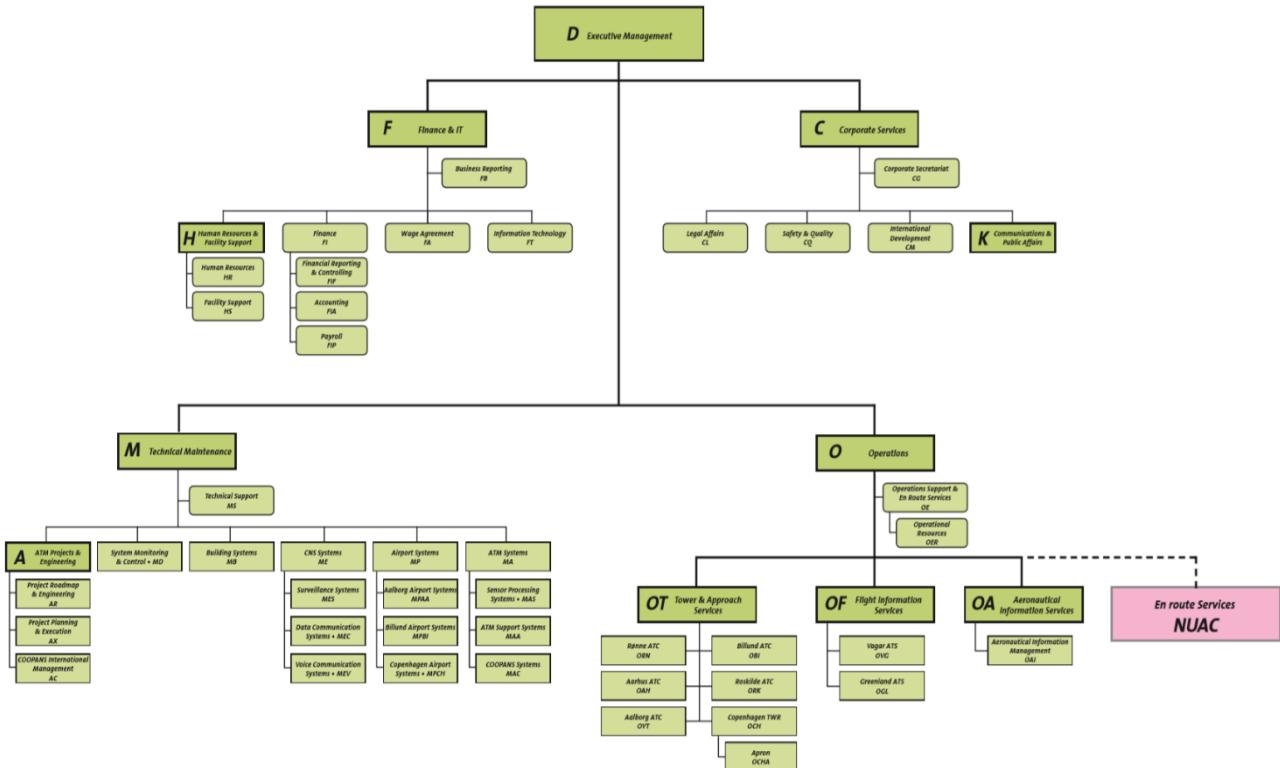
Naviairs organisationsform er som de fleste store virksomheder, bygget op efter linje-stabsprincippet. Dette er grundet at opgaverne i store virksomheder tit har større krav når det kommer til viden. Virksomhederne bliver derfor nødt til at ansætte eksperter og rådgivere inden for nogle bestemte områder. Disse kaldes for stabsmedarbejdere. Stabsmedarbejderne har ingen autoritet til at give ordrer til linjechefen, men er der kun for at rådgive og guide.

Fordelene ved linje-stabsprincippet er at man i virksomhederne får benyttet sig af specialister inden for specifikke områder, og at man er med til at aflaste linjeledere. Ulemper er at der lettere kan opstå konflikter mellem linjelederne og stabslederne, da det er stabsfunktionen der har den konkrete specialistviden, men linjelederen der tager den endelige beslutning.⁷

Dette er et organisationsdiagram over Naviairs opbygning.

⁶ (Naviair - Mission og vision, u.d.)

⁷ (Organisation, 2012)



SWOT – analyse

Interne forhold	
Styrker	Svagheder
<ul style="list-style-type: none"> • Stor viden og ekspertise • Internationalt samarbejde • Vækst i flytrafikken gennem dansk luftrum • Høj medarbejder tilfredshed. • Godt image og omdømme. • Innovativ ledelse. • Meget høj safety og security awareness. • Stabilitet og pålidelighed. 	<ul style="list-style-type: none"> • Lovbestemt løn og regeringsbestemte priser/indtægter - reguleret. • Monopol tilstand (ingen konkurrence) • Meget lange implementeringstider.

Eksterne forhold	
Muligheder	Trusler
<ul style="list-style-type: none"> • Mulighed for at udvide og finde nye teknologier som kan effektivisere deres systemer (Aireon) • Internationalt samarbejde • Ai (Kunstig intelligens) • Big data/Automatisering • Nye teknologier. 	<ul style="list-style-type: none"> • Stort mål for terror og cyberangreb. • Privatisering. • International og EU-regulering/lovgivning.

Styrker

Naviair har som Danmarks eneste luftfartstjeneste masser af års ekspertise og kæmpe erfaring indenfor området. De bliver i business verdenen omtalt med respekt, og de har et godt image og omdømme. Set fra et internationalt perspektiv er Naviair også blandt de bedste. De blev i 2007 kåret som Skandinaviens bedste flyvesikringstjeneste og kom som nummer seks på verdensplan.⁸ Naviair er også blandt de førende, når det kommer til internationalt samarbejde. De har indgået i mange alliance der er med til at effektivere flykontrollen på et større plan. Sammen med de 4 andre europæiske partnere i COOPANS Alliancen vandt de i 2016 EU-Kommisionens "Single European Sky Award". Dette er en award der gives for at anerkende de mange gode tiltag og projekter, som COOPANS Alliancen i fællesskab har startet for at forbedre den europæiske luftfarts kontrol.⁹

Naviair går ind for stabilitet og pålidelighed. De gør utrolig meget ud af at sikkerhedsniveauet er i top, og sørger for at trafikafviklingen er effektiv og forsinkelsesfri (dette har den været siden 2008).

Svagheder

Naviair er som tidligere nævnt en selvstændig offentlig virksomhed ejet af den danske stat. Dette vil sige at de er den eneste leverandør af luftfartstjeneste i Danmarks luftrum, og således uden

⁸ (Naviair - Pressemeldelse, 2007)

⁹ (Naviair - COOPANS, u.d.)

konkurrenter. Dette kan godt være en svaghed, da konkurrence er med til at skabe nytænkning, effektivisering og forbedring af eksisterende systemer.

Det at Naviair er ejet af den danske stat, gør også at de har en fast lovbestemt løn. Det kan være et minus og grund til at nogle fravælger et job i virksomheden. En anden svaghed er de meget lange implementeringstider der er, når der skal tages nye systemer eller udstyr i brug. Grunden til dette er at de skal være 100 procent sikre på at systemet/udstyret fungere som det skal, og snakker ordentligt sammen med resten af virksomhedens systemer, inden de kan skifte over.

Muligheder

Naviairs internationale samarbejde gør også at der bliver åbnet op for nye muligheder inden for teknologien. De ejer sammen med fem andre landes luftfartstjenester selskabet Aireon.

Aireons mission er at få etableret verdens første satellitbaserede fuldt globalt dækkende overvågningssystem til luftfarten. Systemet gør det muligt at hente data om alle fly, hvor som helst på kloden. Det har store sikkerhedsmæssige fordele, og er med til at få udnyttet luftrummet meget mere effektivt end i dag. Systemet skulle gerne være kørende i efteråret 2018.¹⁰

Der er også i fremtiden mulighed for implementering af AI (kunstig intelligens) og automatisering, ved hjælp af big data.

Trusler

Da Naviair arbejder med luftfarttjeneste og flysikkerhed, er en af de største trusler helt klart terror og cyberangreb. Selvom de tager alle tænkelige tiltag, i form af bl.a. lukkede netværker og backup systemer, for at undgå at netop dette kommer til at ske, er det stadig en kæmpe trussel.¹¹

En anden trussel for virksomheden er, at de skal rette sig ind under EU's regulering og lovgivning. Det går fint lige nu, men det kan måske i fremtiden blive et problem at opfylde deres lovgivning, hvilket kan give problemer.

¹⁰ (Naviair - Aireon, u.d.)

¹¹ Fremgik af mødet med Naviair

Feasibility study

Inden vi kommer for godt i gang med selve design – og programmeringsdelen, skal vi først have lavet en feasibility study. Dette gør vi for at vise sandsynligheden for at projektet bliver en succes, og for at finde de eventuelle svagheder der nu skulle være ved projektet, og hvordan vi for dem klaret.

Der er nogle faktorer der ikke rigtig spiller ind i forhold til vores projekt, som blandt andet de økonomiske og finansielle forhold. Normalt ville man vurdere om et færdigt projekts indtjeninger i sidste ende ville overstige omkostningerne, før man godkender det. Men da vi laver dette projekt som en del af vores eksamensopgave i vores fritid, og derfor ikke skal betales, er dette ikke væsentligt at snakke om i sammenhæng med vores projekt. Med hensyn på de retslige og juridiske forhold, gør vi i vores projekt ikke brug af noget af Naviairs rigtige data. Dette ville nemlig være en overskridelse af deres sikkerhedsprocedure. Vi arbejder kun med "uægte" informationer om deres it-udstyr, og hvad de indeholder, i vores projekt.

Tekniske forhold

Systemet vi skal have lavet, skal kunne skabe overblik over det forskellige it-udstyr der er i Naviair. Det skal kunne vise det bestemte udstyrts service og hvilke applikationer det indeholder. Der skal være to brugere i form af en admin og en helpdesk-bruger. Administratoren skal kunne håndtere udstyr, service, applikationer og brugere i systemet. Helpdesk-brugeren skal kunne se udstyr, service, applikationer og deres forskellige relationer.

Da vi selv skal stå for hele den tekniske proces, er der i dette her tilfælde ikke brug for at outsource til eksterne og mere kompetente underleverandører.

Projektet er helt sikkert opnåeligt, men da vi kun har 25 dage til at få systemet op at køre, er det meget muligt at vi ikke kan aflevere et helt færdigt produkt til Naviair. De vil dog stadig i sidste ende få en rigtig god fornemmelse af, hvordan et færdigt produkt ville kunne gavne og implementeres ind i deres virksomhed, hvis de skulle ønske at gå videre med projektet.

Organisatoriske og operationelle forhold

Vores projekt er blevet udarbejdet i samarbejde med Naviairs security manager, og vi ved derfor ikke om Naviairs formand skulle have en reel interesse i det endelige produkt. Vi mener dog bestemt at vores løsning om et arkivsystem ville være til gavn og en fordel for Naviair.

Systemet ville højne serviceniveauet og gøre helpdesks arbejde mere effektivt og brugervenligt, da der ville være lettere adgang til de informationer der søges. Skulle vores system ende ud med at blive implementeret i virksomheden, ville vi helt sikkert lave et testhold af de davaerende helpdesk arbejdere. De ville kunne hjælpe os med at få bygget et system, der har en brugergrænseflade der er nem at forstå, og tilpasset til deres behov. Ved at involvere dem, ville man på den måde forhåbentlig gøre implementeringsprocessen lettere.

Interessent analyse

I en opgave som denne er det også rigtig vigtigt at finde ud af hvem det er der har interesse i projektet, og mulige modstandere. Måden vi finder ud af det på, er ved at lave en interessent analyse. Den er med til at give et fuldt overblik over hvem der har indflydelse på projektet, og hvordan man skal håndtere eventuelle forekomne problemer.

Der vil normalt være fire forskellige typer interesserter, når man laver et projekt.

Ressourcepersonalet er den gruppe, der både har indflydelse og bliver påvirket af projektets udkom. Dette er derfor også den vigtigste interessentgruppe.

Grå eminence gruppen er egentlig ikke superinteresserede i selve projektet da det ikke nødvendigvis påvirker dem, men de har stadig en stor indflydelse på om det bliver til noget. Gidsel gruppen har tit ikke særlig megen indflydelse på projektet, men bliver til gengæld meget påvirket af det endelige resultat.

Den sidste interessentgruppe er de eksterne interesserter. Denne gruppe har hverken indflydelse eller påvirkning på hvordan projektet ender ud. De skal kun meddeles at projektet eksisterer.

Vi har valgt at lave to skemaer. Det ene viser interessenternes relationer til projektet, og det andet giver et bedre overblik over de vigtigste interesserter og deres betydning for projektet.¹²

	Stor indflydelse	Lille indflydelse
Bliver påvirket af projektet	Formand	Administrator Helpdesk Naviair medarbejdere
Bliver ikke påvirket af projektet	Security Manager	

¹² (Organisation, 2012)

Interessent	Fordele	Ulemper	Vurdering	Håndtering
Naviairs formand	Effektivisering af deres nuværende system, hvilket også gør det billigere og kan spare virksomheden penge på sigt.	Da vores projekt ikke koster Naviair nogen penge, men blot er et "mock-up" system de kan gøre brug af eller lade være, er der ikke nogen ulemper.	Alt i alt er projektet en fordel for Naviairs formand.	Formanden skal informeres om det færdige produkt/system, og være en del af beslutningerne.
Admin	Får et bedre overblik over al firmaets it-udstyr, med en nem og ligetil håndteringsproces af udstyr, services og applikationer.	Skal lære et nyt system at kende. Har ikke haft mulighed for at være en del af udviklingsprocessen og kan derfor ses lidt som en "gidsel".	Da admin får en nemmere håndteringsproces og vi forsøger at gøre systemet så brugervenligt så muligt, er det alt i alt en fordel.	Ingen håndtering
Helpdesk	Sparer tid og ressourcer, og får derfor mere tid til andet arbejde	Skal lære et nyt system at kende. Har ikke haft mulighed for at være en del af udviklingsprocessen og kan derfor ses lidt som en "gidsel".	Helpdesks arbejde bliver gjort nemmere og vi forsøger at gøre systemet så brugervenligt så muligt, så alt i alt er det en fordel.	Ingen håndtering
Naviair security manager	Da det ikke er security manageren der skal gøre brug af systemet, men primært Helpdesk og Admin, bliver han ikke påvirket af det nye system som sådan.	Skal lære et nyt system at kende, men da han selv er inde over udviklingen, og derfor selv kan præge designet, er det ikke en stor ulempe.	Systemet kommer ikke til at ændre hans arbejdsproces, og er derfor ikke super relevant for ham.	Skal inkluderes i hele processen og være en del af beslutninger og opdateringer.

Risikoanalyse

Vi skal også have lavet en risikoanalyse inden vi kommer for godt i gang med selve projektet. Grunden til at vi laver denne, er for at finde de alvorligste risici i forhold til gennemførelsen af projektet. Både for at finde løsninger, men også for at bestemme hvem der har ansvar for at mindske og overkomme disse risici.

Man skulle i sidste ende gerne stå tilbage med et klart billede af, hvordan man ville håndtere eventuelt opståede problemer.

Her er nogle af de risikomomenter vi kunne finde i forbindelse med vores projekt.

Risiko moment	Sand-synlighed (1-5)	Konsekvens (1-5)	Produkt (S x K)	Handlinger (forebyggende + afbødende)
Projektet bliver afbrudt	1	5	5	Sikre at der er et godt samarbejde mellem kunde (Naviair) og udviklere.
Ændringer i krav Specifikationerne	2	5	10	Grundig gennemgang af alle kravspecifikationerne med kunden inden der bliver startet på udvikling.
Forsinkelser i levering af det endelige system	4	2	8	Brug tid på at estimere og planlægge opgaven korrekt i inception-fasen.

Glossary

Equipment / udstyr	Forskellig it-udstyr såsom tablet, bærbar, pc, mobil, skærm osv.
Services	Forskellige services som it-udstyret kan udføre, som fx it-udstyr kan bruges til flyplaner eller har radar access
LoginMenu	<ol style="list-style-type: none"> 1. Admin login 2. Helpdesk login
helpdeskMenuShow	<ul style="list-style-type: none"> • Read Equipment • Read Service • Read Application • Read Equipment_type • Read Manufacturer • Logout
adminMenu	<ul style="list-style-type: none"> • Create Equipment • Create Service • Create Application • Create Equipment_type • Create Manufacturer • Read Equipment • Read Service • Read Application • Read Equipment_type • Read Manufacturer • Update Equipment • Delete/Archives Equipment • Delete/Archives Service • Delete/Archives Application • Delete/Archives Equipment_type • Delete/Archives Manufacturer • Logout

Admin	Administrator
Helpdesk	IT support medarbejder som kan hjælpe IT-brugere i virksomheden.
Apps/Application	Applikationer
Modal box	Et vindue der ikke sender videre til en ny side, men popper op på den daværende side.

Unified process

Vi har udarbejdet vores opgave ved hjælp af unified process metoden. Dette har gjort at vores system har ændret sig meget fra starten til slutningen af projektet. Der er både blevet fjernet og tilføjet funktioner, da vi undervejs i processen langsomt fik en bedre forståelse for hvad det var Naviair havde brug for.

Da der for det meste popper uforudsigelige problemer op hen ad vejen, giver UP metodens iterationer mulighed for at gå tilbage at kigge på designdelen igen og få den tilrettet, således at man opnår det bedst mulige resultat.

Vi har delt inception-fasen, elaboration-fasen og construction-fasen op, sådan så man får en bedre ide om hvad vi har lavet og hvad der er blevet ændret undervejs i forløbet.

Inception-fasen

Systemvision

Dette system skal kunne skabe overblik over forskellige it-udstyr der er i Naviair.

Det skal kunne vise det bestemte udstyres service og applikationer. Administratoren skal kunne håndtere udstyrer, service, applikationer og bruger i systemet. Helpdesk skal kunne se udstyrer, service, applikationer og deres forskellige relationer.

Kravanalyse

Vi var så heldige at vi fik lov til at komme ud ved Kastrup lufthavn og besøge Naviair. Vi fik først en rundvisning af virksomheden, hvor vi blandt anden var oppe i kontroltårnet, og nede i

serverrummet. Bagefter interviewede vi Naviairs security manager for at finde virksomhedens mål, kravene til det færdige system, og de ansattes roller i forhold til systemet.

Vores opgave er at lave et arkivsystem over alt Naviairs IT-udstyr. Der skal stå oplysninger om hvad udstyret indeholder af applikationer og hvilke services det kan yde. Vi kan af sikkerhedsmæssige årsager ikke få virksomhedens data om det IT-udstyr de ligger inde med, og vi bruger derfor i denne opgave og i vores system selvopfundne data.

Hjemmesiden skal kunne afspejle virksomhedens identitet. Det er derfor vigtigt, at brugeren ved, at det er en hjemmeside for en virksomhed der er baseret på sikkerhed. Derfor skal adgangen til diverse informationer være tydelig gjort.

Desuden skal systemet opfylde målgruppens forventninger og behov.

Målet med systemet er altså en over all effektivisering af deres aktuelle arbejdsmetode, som skal være overskueligt og let at gå til. Det skal være nemmere at søge informationer og håndtere dem, og i sidste ende formindske tidsforbrug og resurser.

Funktionel/ikke-funktionel

Equipment		F/IF
K1	Systemet skal kunne vise Naviairs it-udstyr	F
K2	Systemet skal kunne registrere it-udstyr	F
K3	Systemet skal kunne redigere eksisterende it-udstyr	F
K4	Systemet skal kunne slette enkelte it-udstyr	F
Application		
K5	Systemet skal kunne vise it-udstyrets applikationer	F
K6	Systemet skal kunne registrere it-udstyrets applikationer	F
K7	Systemet skal kunne redigere it-udstyrets applikationer	F

K8	Systemet skal kunne slette it-udstyrets applikationer	F
Service		
K9	Systemet skal kunne vise Naviairs services	F
K10	Systemet skal kunne registrere services	F
K11	Systemet skal kunne redigere eksisterende services	F
K12	Systemet skal kunne slette services	F
User		
K13	Systemet skal kunne oprette, redigere og slette brugere	F
K14	Admin skal kunne læse, søge, registrere, redigere og slette i systemet	F
K15	Helpdesk skal kun kunne læse og søge i systemet	F
K16	Alt indhold på websitet skal ikke være offentligt tilgængelig for alle. Dvs, at der skal være loginfunktion på websitet.	F
Miscellaneous		
K17	Systemet skal demonstreres på en public cloud såsom AWS	F
K18	Systemet skal laves i Java	IF
K19	Systemet skal kunne køre via en web browser (Google Chrome fra version 60.x eller nyere, Internet Explorer 10 eller nyere, Firefox)	F
K20	Der skal være dokumentation som beskriver hvordan systemet er bygget	IF
K21	Der skal vedlægges en brugervejledning	IF

K22	Designet på hjemmesiden skal være brugervenligt, ikke for mange skarpe farver.	IF
K23	Siden skal have Naviairs logo så det tydeligt indikere at man bruger en Naviair side.	F
K24	Sprog. Det generelle sprog på websitet skal være engelsk.	F
K25	Systemet skal have en oppetid på mindst 99,9%	F
K26	Systemet skal have en responstid på max 2 sekunder pr. side.	F

OBS: Der kan forekomme ændringer i Elaboration-fasen

FURPS

Functionality

Den elementære funktion i vores system er at give brugeren en smart og hurtig måde at udføre håndtering af it-udstyr, applikationer og services. Med håndtering menes der visning, tilføjelse, redigering og slettelse. Det er også et krav at der er en søgefunktion der gør det let at søge i systemet.

Usability

Systemet skal laves i fællesskab med Naviairs security manager. Hans feedback skal gøre at vores system ender ud med at blive ligesom Naviairs ønsker det. Vi har stillet dem en masse spørgsmål om designet, for at kunne tilpasse brugergrænsefladen til medarbejdernes behov. De ønsker en simpel brugergrænseflade som er nem at navigere. Der skal være ikoner for *create* og *search*, for at opnå et mere sleek udseende. Farvede knapper skal også gøre systemet nemmere at navigere. Der skal vedlægges en brugervejledning der forklarer hvordan det endelige system navigeres.

Reliability

Vi vil i vores system sørge for at undgå redundant data. Dette kan nemlig gøre systemet langsommere og opbruge unødvendig plads i databasen. Måden vi gør det på, er ved f.eks. at gøre sammensætningen af *equipment name* og *equipment serial number* unique i MySQL. På den måde er vi sikre på at Admin ikke ved en fejl, kan tilføje det samme udstyr flere gange.

Dette skal mindske muligheden for fejl i databasen, og uventede systemfejl.

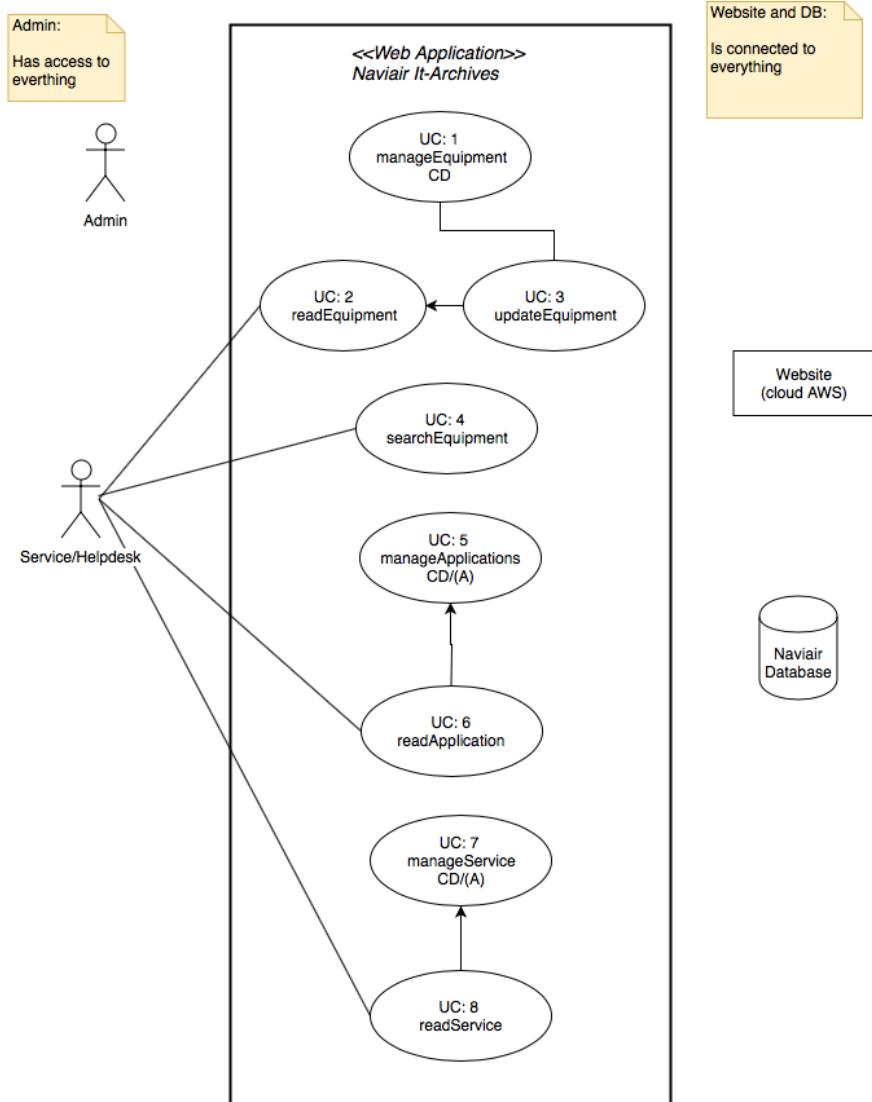
Performance

Da Naviairs krav til systemet er en oppe tid på 99,9% og en responstid på max 2 sekunder, er det super vigtigt at vi får lavet et system som ikke indeholder en masse overflødige data og kode. Der skal kun være den mængde, der akkurat er brug for, for at systemet fungerer. Således at brugeren med nogle enkelte klik, kan finde frem til det ønskede mål.

Supportability

Det generelle sprog på websiden skal være engelsk, da Naviair har ikke-dansk-talende medarbejdere.

Use case model



OBS: Der kan forekomme ændringer i Elaboration-fasen

Aktørbeskrivelser:

Admin. (Administrator)

Da programmet "bare" skal være en oversigt over det forskellige it-udstyr der er i Naviair, skal systemet være simpelt bygget op. Systemet skal være brugervenligt og nemt at håndtere, da det skal bruges til overblik. Admin skal kunne tilføje, læse, redigere og slette udstyr, services, applikationer og brugere.

(Bruger har adgang til systemet)

Helpdesk(Service)

Helpdesk er ikke så omfattende som admin, da helpdesk kun skal kunne se det forskellige udstyr, services og applikationer.

Domæne model

Vi har lavet vores domænemodel som en repræsentation af meningsfulde virkelige ting, vi mente var relevante for dette domæne. Vi lavede modellen for at vise kunden hvilke ting vi skulle arbejde med og hvilke relationer tingene havde til hinanden. Modellen blev brugt som kommunikationsværktøj mellem kunde og os som udviklere, og til at få en bedre forståelse af deres ordforråd. Vi kan som udviklere senere hen også bruge domænemodellen som en guideline til at modellere vores software. Da vi er to til at udvikle programmet, gav det os også en klarhed og tydelige rammer over programmets opbygning. Denne domænemodel indeholder otte konceptuelle klasser.

Equipment som indeholder it-udstyr og dets information.

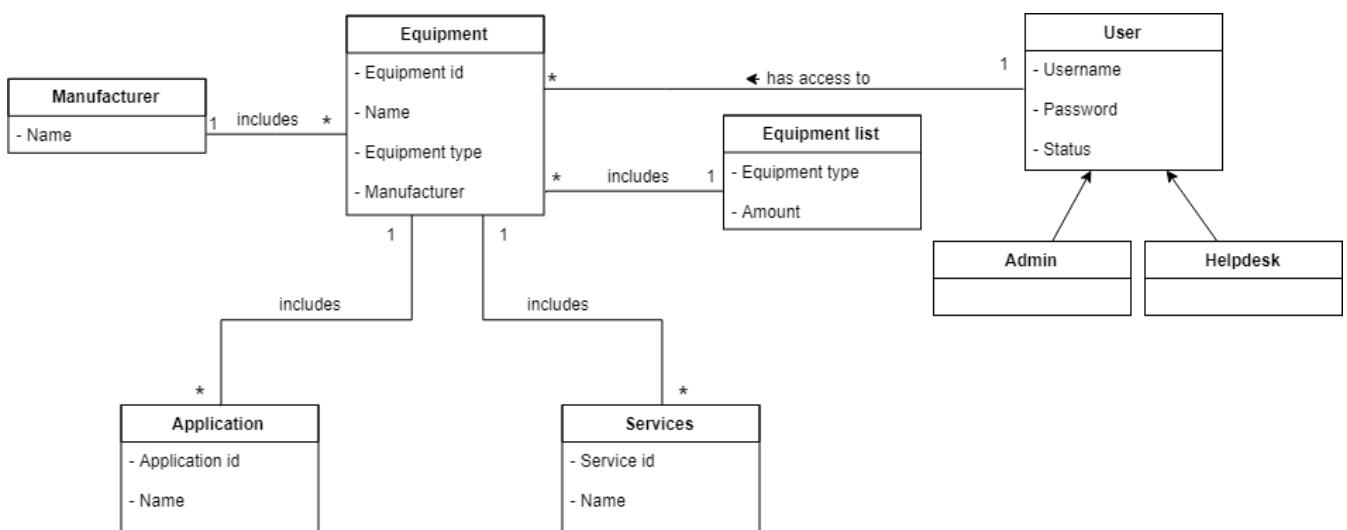
Application som indeholder diverse applikationer og deres information.

Service som indeholder diverse ydelser (Mail, Radar osv.) og deres information.

Manufacturer hvilket er en liste over alle de forskellige producenter.

Equipment list hvilket er en liste over det forskellige slags udstyr.

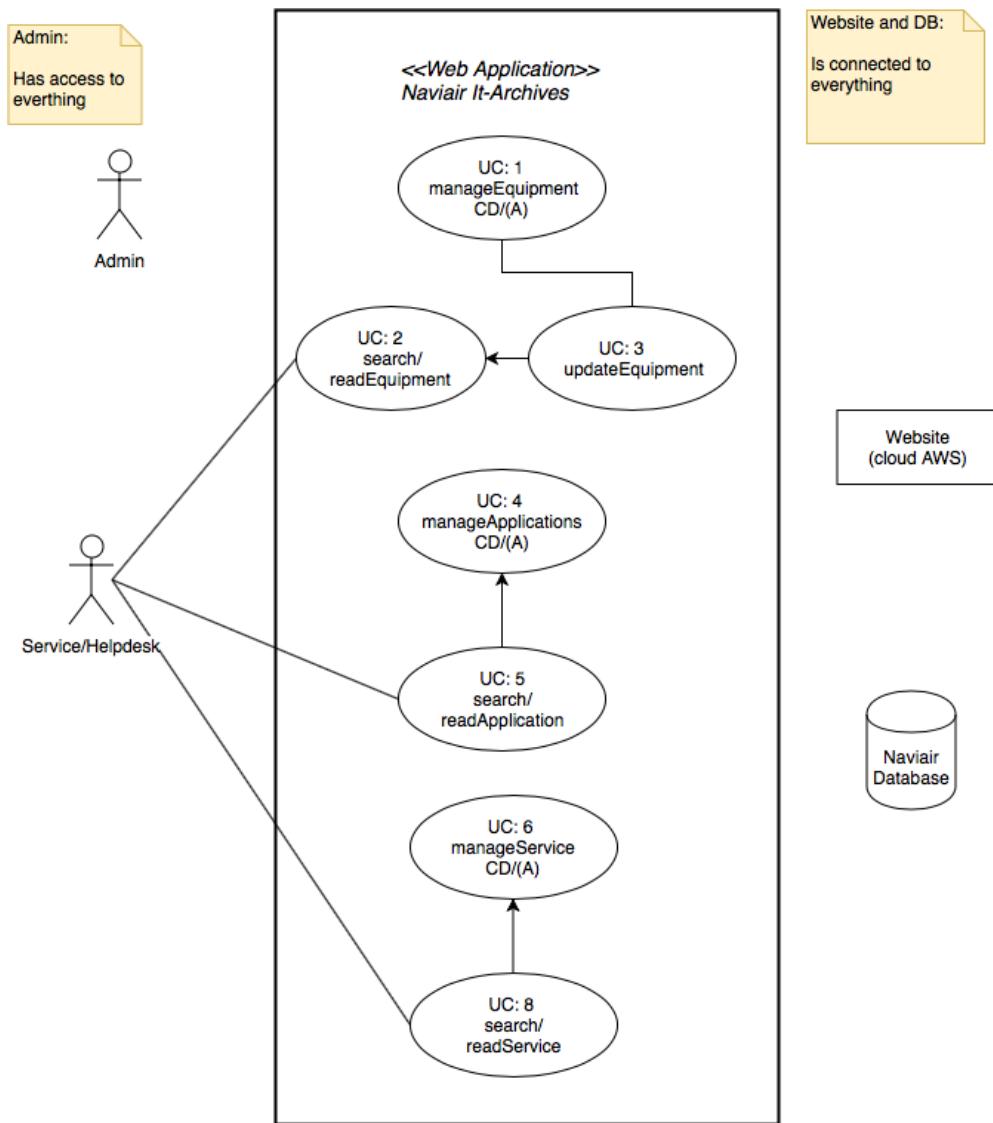
User, som er en bruger der kan logge ind i systemet. Her er to forskellige roller, Admin og helpdesk, som har adgang til forskellige ting.



OBS: Der kan forekomme ændringer i Elaboration-fasen

Elaboration-fasen

Use case model



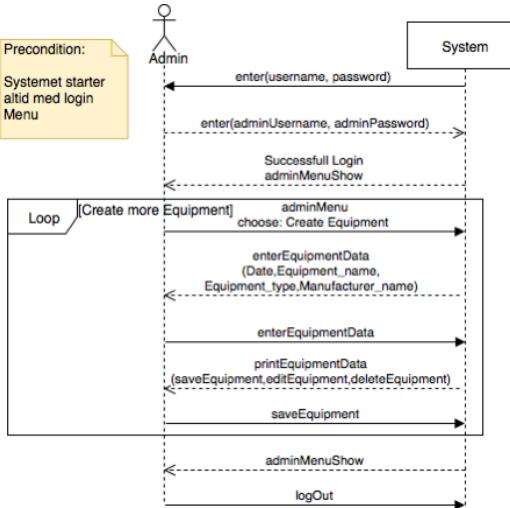
Dette er blevet ændret i Use case-modellen fra Inception-fasen:

Inception-fasen UC 4(searchEquipment) er blevet slettet, og searchEquipment er i stedet for blevet koblet på UC2: *readEquipment*, da read og search funktionen minder meget om hinanden.

System sekvensdiagram

Dette er et system sekvens Diagram over første del af UC1: *manageEquipment*. Det viser hvordan admin interagerer med systemet, og de systemhændelser som admin genererer, ved håndtering (i dette tilfælde tilføjelse) af udstyr. Vi har lavet dette diagram for at illustrere main succes scenario

på input og output hændelser mellem admin og systemet, og for at definere den ønskede system adfærd.

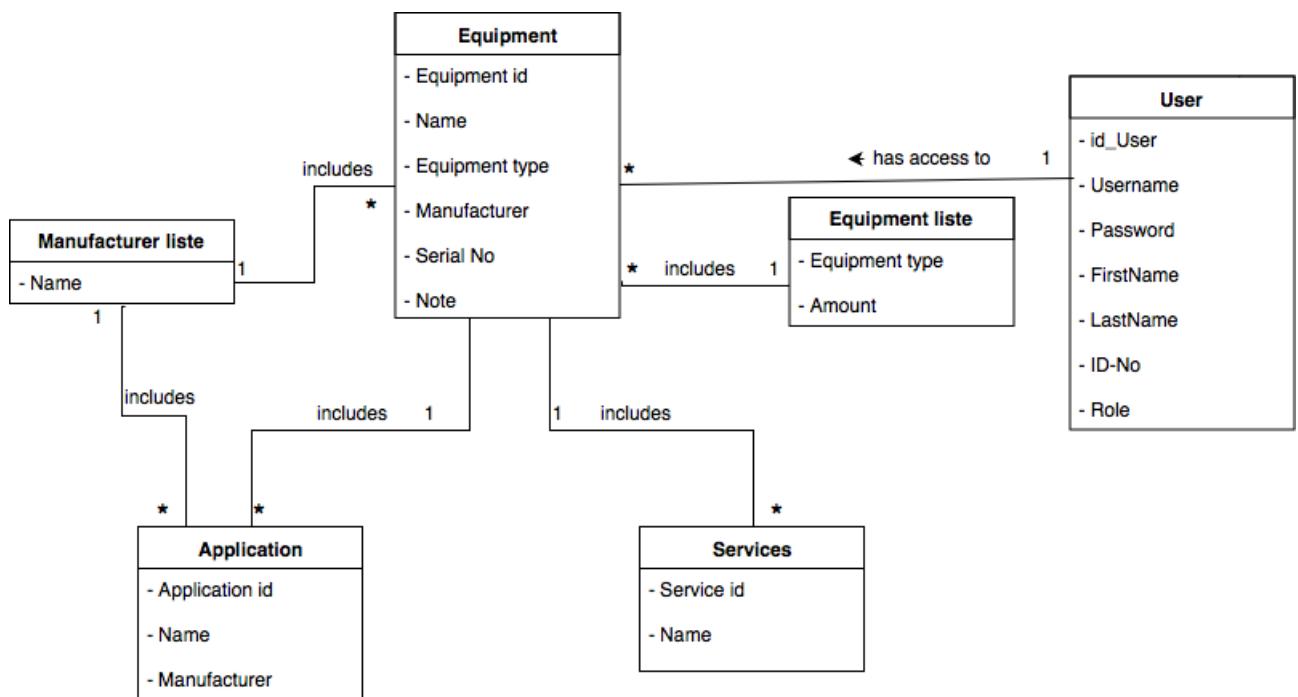


Domæne model

Følgende er ændret siden inception-fasen:

Vi har fjernet de to under-roller der var under User og i stedet for kun en bruger.

Application har fået en manufacturer.



Fully dressed use cases

UC1: manageEquipment

Vi har som det første valgt at lave en fully dressed use case af UC1 (manageEquipment). Grunden til vi har valgt denne er at vi føler at tilføjelse (og sletning) af it-udstyr, er en de fundamentale funktioner som vores system skal kunne udføre. Derfor er det vigtigt at vi kommer mere i dybden i lige præcis denne, og finder så mange detaljer som muligt. Dette vil også gøre det lettere når den til sidst skal kodes.

UC1: manageEquipment

Scope: IT arkivsystem

Level: User goal

Primary actor: Admin

Stakeholders and interests:

- **Admin:** Vil kunne tilføje IT-udstyr nemt, hurtigt og uden fejl til databasen så det kan findes i arkivsystemet.
- **Firmaet:** Vil gerne have oversigt over alt det IT-udstyr de har til rådighed. Vil kunne se hvilke services hvert enkelt IT-udstyr har at tilbyde. Vil kunne se hvilke apps der er installeret på det forskellige IT-udstyr. Vil kunne se hvornår hvert enkelt IT-udstyr er blevet tilføjet til arkivsystemet.
- **Helpdesk:** Vil kunne se det nye IT-udstyr og finde det ved hjælp af søgefunktionen i arkivsystemet.

Preconditions: Admin er logget ind i systemet.

Success guarantee: Admin har succesfuldt tilføjet et nyt IT-udstyr til databasen og det er nu søgbart i arkivsystemet.

Main success scenario:

1. Firmaet modtager nyt IT-udstyr
2. Admin logger ind i arkivsystemet
3. Admin trykker på  i toppen af venstre hjørne.
4. Systemet spørger om equipment name, date, equipment type og manufacturer name.

5. Admin skriver de korrekte oplysninger ind i de rette felter.
6. Admin trykker på create knappen.
7. Systemet tilføjer det nye IT-udstyr til databasen, og tildeler det automatisk et unikt ID.

Extensions:

*a. Hvis systemet går ned/fejler

1. Admin genstarter browseren og prøver igen.
2. Hjælper det ikke, kontakter Admin udviklerne, som hjælper med at løse problemet.

2a. Admin kan ikke logge ind i systemet.

1. Systemet informerer Admin om at username eller password er forkert.
2. Admin forsøger at logge ind igen.
 - 2a. Hvis Admin stadig ikke kan huske rette username og password, må han kontakte udviklerne.
 - 2b. Admin får hjælp til at resette hans password, og kan nu logge ind.

5a. Admin skriver de forkerte oplysninger ind.

1. Admin finder IT-udstyret med de forkerte oplysninger.
2. Admin trykker på edit funktionen udfør IT-udstyret.
 - 2a. Admin skriver de rigtige oplysninger ind og gemmer.
3. IT-udstyret er nu gemt i databasen igen med de rette oplysninger.

7a. Systemet fejler i at gemme det nye IT-udstyr, og det kan ikke ses i databasen.

1. Admin starter forfra og forsøger at tilføje IT-udstyret igen.
2. Hvis systemet fejler igen, må Admin kontakte udviklerne.
3. Udviklerne tjekker at databasen er rigtigt forbundet med systemet.
 - 3a. Hvis den ikke er det, sørger de for at forbinde dem korrekt.

Special requirements:

- Løsningen kan kører via en webbrowser, såsom Google Chrome fra version 60.x eller nyere, Internet Explorer 10 eller nyere, Firefox.

- Systemet skal være ligetil og brugervenligt – dette opnås ved hjælp af en enkel brugeroverflade med farver der gør det nemt at navigere.

Technology and data variations list:

Løsningen skal laves i Java.

Løsningen skal demonstreres på public cloud som AWS.

Frequency of occurrence:

Det er svært at give et præcist estimat af hvor meget IT-udstyr der bliver tilføjet til systemet, men det er i gennemsnittet omkring 5-10 nye apparater om måneden.

UC2: searchEquipment

Den anden use case vi har valgt at lave en fully dressed af, er UC2 (searchEquipment). Dette er også en af de basale use cases, som har en stor vigtighed i vores færdige system. Det er denne funktion der skal hjælpe med at effektivisere Helpdesks arbejde, og egentlig også Admins, da sletning og redigering af et specifikt IT-udstyr, også går hurtigere når Admin blot kan søge efter det i systemet.

Scope: IT arkivsystem

Level: User goal

Primary actor: Helpdesk

Stakeholders and interests:

- **Helpdesk:** Vil kunne se alt Naviairs IT-udstyr og finde det ved hjælp af søgefunktionen i arkivsystemet.
- **Admin:** Vil kunne se alt Naviairs IT-udstyr. Han skal også bruge søgefunktionen i arkivsystemet til hurtigt at finde et specifikt stykke udstyr, hvis han skal slette eller rette det.
- **Firmaet:** Søgefunktionen hjælper med at effektivisere deres hverdag, da helpdesk medarbejderne hurtigere kan finde frem til det ønskede it-udstyr. Dette sparer dem tid og

ressourcer.

Preconditions: Helpdesk er logget ind i systemet.

Success guarantee: Helpdesk har succesfuldt fundet frem til et specifikt IT-udstyr i arkivsystemet.

Main success scenario:

1. Helpdesk logger ind i arkivsystemet.
2. Helpdesk trykker på  i menuen.
3. Der popper en modal box op, og her er der fire søgefelter til henholdsvis equipment ID, equipment type, service og applikationer.
4. Helpdesk finder ud af hvilket felt han skal søge i, og skriver de ønskede søgeoplysninger ind.
5. Helpdesk trykker på *search* knappen.
6. Systemet viser det it-udstyr der er relateret til det indtastede søgeord.

Extensions:

*a. Hvis systemet går ned/fejler

1. Helpdesk genstarter browseren og prøver igen.
2. Hjælper det ikke, kontakter Helpdesk Admin, som hjælper ham/hende med at få løst problemet.

1a. Helpdesk kan ikke logge ind i systemet.

1. Systemet informerer Helpdesk om at der er skrevet forkert username eller password.
2. Helpdesk prøver på at logge ind igen.
 - 2a. Hvis Helpdesk stadig ikke kan huske hans/hendes korrekte username og password, må han kontakte Admin.
 - 2b. Admin hjælper Helpdesk med at resette hans/hendes username og password, og Helpdesk kan nu logge ind.

4-6a. Helpdesk skriver oplysningerne ind i det forkerte søgefelt.

1. Systemet siger at det søgte, ikke findes i arkivsystemet.
2. Systemet foreslår at man tjekker om man har skrevet søgeordet ind i det rette søgefelt og foreslår at man prøver igen.
 - 2a. Helpdesk skriver de rigtige oplysninger ind og finder nu frem til det søgte udstyr.

Special requirements:

- Løsningen kan kører via en webbrowser, så som Google Chrome fra version 60.x eller nyere, Internet Explorer 10 eller nyere og Firefox.
- Systemet skal være ligetil og brugervenligt – dette opnås ved hjælp af en enkel brugeroverflade med farver der gør det nemt at navigere websiden.

Technology and data variations list:

Løsningen skal laves i Java.

Løsningen skal demonstreres på public cloud som AWS.

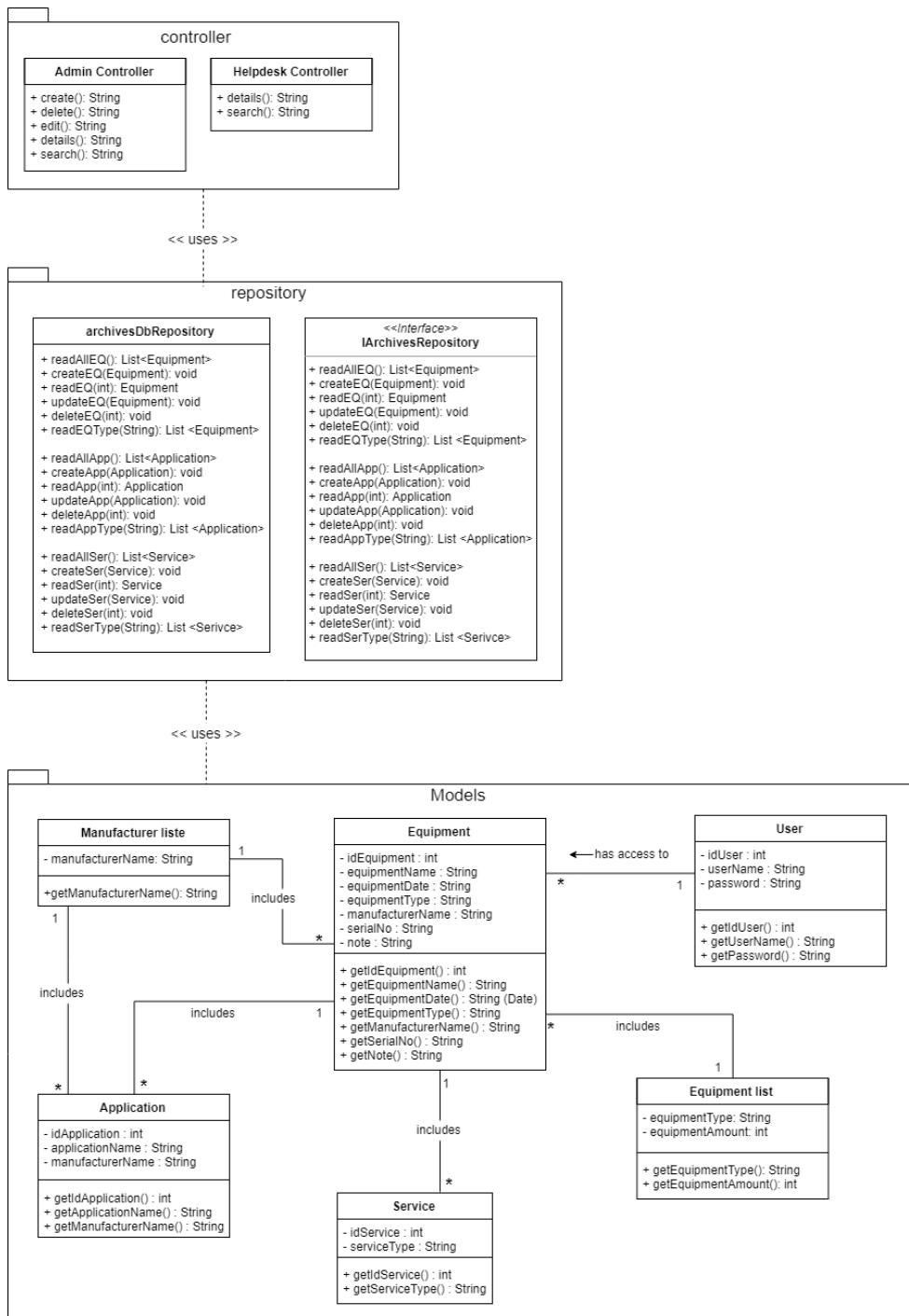
Frequency of occurrence:

Det er umuligt at give en helt akkurat vurdering af hvor tit systemet vil blive brugt, men kunden regner med at det nok er op til flere gange dagligt.

Construction-fasen

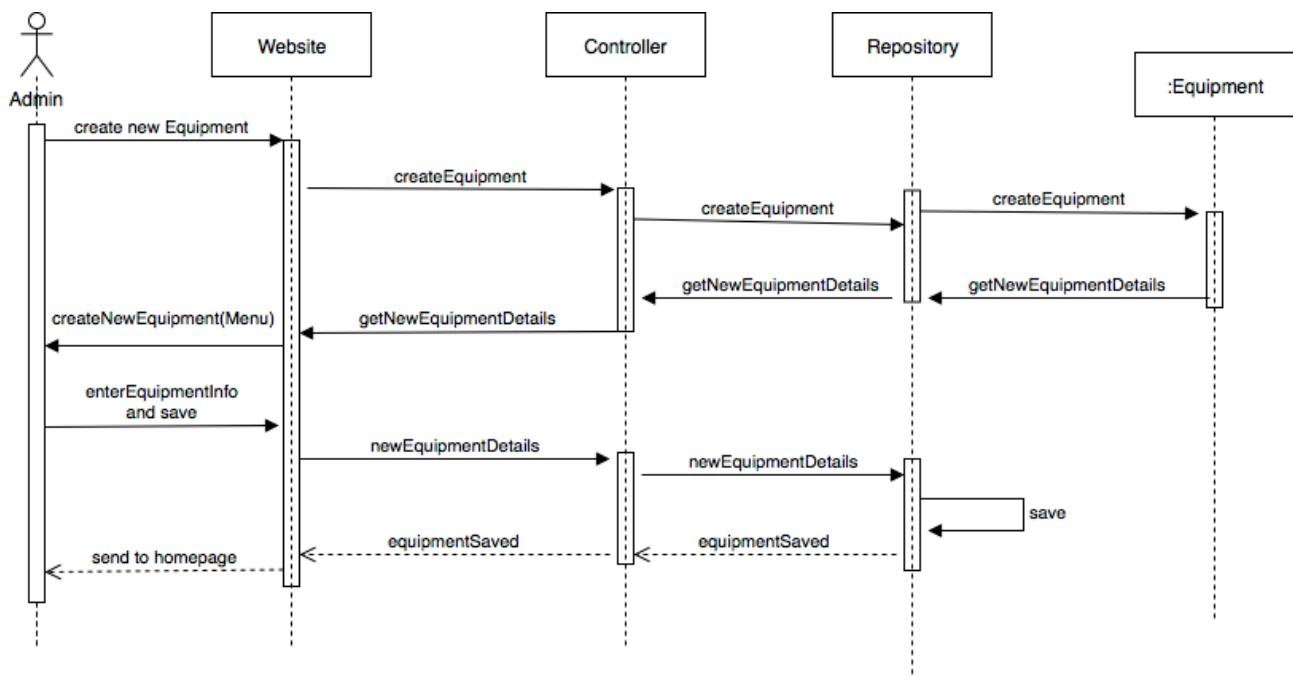
Klassediagram

Vi har valgt at lave et klassediagram, da det er med til at give os en bedre idé af hvordan vores system skal struktureres når vi skal begynde at kode. Vi får her styr på forbindelsen mellem klasserne, arv og hvilke attributter og metoder de hver især skal have. Klassediagrammet afspejler ligesom domænemodellen det meningsfulde håndgribelige liv.



SD diagram

Vi laver et sekvensdiagram for at få beskrevet de interaktioner der er mellem de forskellige klasser i et dynamisk forløb. Det er en god måde at visualisere og validere forskellige use case forløb på, og kan være med til at forudsige, hvordan et system vil fungere. Det kan også hjælpe til med bedre at forstå hvor ansvaret skal ligge henne, i de forskellige klasser.



Database

Vi benytter MySQL til at lagre vores oplysninger/data fra Naviairs arkiv hjemmeside. Det data databasen indeholder er diverse IT-udstyr informater, og de brugere som har adgang til hjemmesidens informationer. Vi har valgt at lave en relations-database, fordi det er nemt at gemme, sortere, finde, ændre, læse og kombinere data ved brugen af denne database.

Arkivet skal indeholde diverse IT-udstyr oplysninger, såsom serienummer, navnet på udstyret, producenten, hvilken service den yder og applikationer den indeholder.

Vores opgave er at komme med et forslag til, hvordan et arkiv over diverse IT-udstyr og deres ydelser, kan komme til at se ud. Det er ikke alle ansatte i Naviair der vil have adgang til dette IT-udstyr arkiv. Der skal være en Admin – og Helpdesk-bruger. Administratoren vil have adgang til at tilføje, rette, se og slette data. Helpdesks kan kun se data, de skal derfor have forskellige logins.

Målet med databasen er, at Naviair får et overblik over de forskellige former for IT-udstyr de har til rådighed, og at man som ansat kan gå ned til Helpdesk og få hjælp til at finde det udstyr man skal bruge.

Beskrivelse af tabellerne

Vores database indeholder syv forskellige tabeller: Equipment, Equipment type, Manufacturer, Application, Applicationlist, Service og Servicelist.

Equipment tabellen viser informationer om IT-udstyret og indeholder attributterne: equipment id, equipment name, manufacturer name, equipment type, equipment date, equipment serial number og equipment note.

Her er equipment id primær nøgle, og sammensætningen af equipment name og equipment serial number er unique, for at undgå at det samme udstyr ved en fejl, bliver tilføjet i databasen flere gange. Equipment har en mange til mange relation til Service tabellen, og vi opretter derfor en Servicelist tabel for at forhindre dette. Der er også en mange til mange relation mellem Equipment og Application, så vi opretter også en tabel der hedder Applicationlist.

Service tabellen viser alle de services Naviair har til rådighed og indeholder attributterne: service id og service type. Service id er her primær nøgle.

Servicelist tabellen viser en liste over hvilke IT-udstyr der indeholder hvilke services. Tabellen indeholder attributterne: service id og equipment id som begge er foreign keys, der peger henholdsvis på Service og Equipment.

Application tabellen viser informationer om alle de forskellige applikationer et IT-udstyr kan have. Tabellen indeholder attributterne: application id, application name og manufacturer name. Application id er i dette tilfælde primær nøgle. Manufacturer name har en foreign key forbindelse til Equipment.

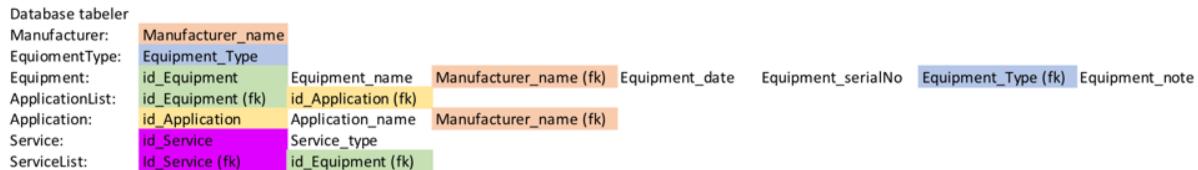
Applicationlist tabellen viser en liste over hvilket IT-udstyr der indeholder hvilke applikationer. Tabellen indeholder attributterne: application id og equipment id, som begge er foreign keys, der peger på henholdsvis Application og Equipment.

Manufacturer tabellen indeholder attributten manufacturer name, som er primær nøgle. Den har en en-til-mange-relation til Equipment.

EquipmentType tabellen viser informationer om hvilken type IT-udstyret er (fx tablet, pc eller server) og indeholder attributten equipment type, som er primær nøgle. Den har en en-til-mange relation til Equipment.

Her er der et skema der daner et overblik over de forskellige tabeller og deres indhold.

Farvekoden er for at fremhæve de forskellige tabellers relationer:



Normalisering

1. Normal form (1-N)

Et IT-udstyr kan have mange services og applikationer. I Equipment-tabellen i databasen har hvert IT-udstyr en primær nøgle, som er kaldet id_equipment. Dette er en oplagt nøgle at bruge som foreign key (FK), da alle diverse IT-udstyr har forskellige id_equipment numre. Vi har sat id_equipment til at være auto incremented (AI), for at sikre at to forskellige IT-udstyr ikke har ens id_equipment numre.

#	Field	Schema	Table	Type
1	id_Equipment	naviairs	equipment	INT
2	Equipment_name	naviairs	equipment	VARCHAR
3	Equipment_date	naviairs	equipment	VARCHAR
4	Equipment_type	naviairs	equipment	VARCHAR
5	Manufacturer_name	naviairs	equipment	VARCHAR
6	Equipment_serialNo	naviairs	equipment	VARCHAR
7	Equipment_note	naviairs	equipment	VARCHAR

Ud fra Equipment tabellen ses det, at der i vores tabel ikke er nogle linjer der indeholder samme eller gentagende data

2. Normal form (2-N)

For at vi skal kunne arbejde med 2. normal form skal tabel opfylde 1. normal form. Vi har udført 2. Normal form i vores database ved at de informationer der står i tabellen, kun hører til den ene primærnøgle.

Så for at undgå at tabellen ser sådan her ud:

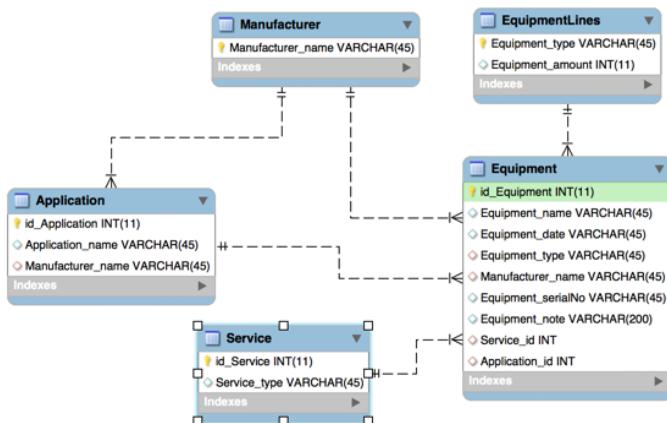
Id_App	App_name	Id_Equip	Equip_name	date	Manufacturer	type	Serial no

Deler vi den op så den ser sådan her ud:

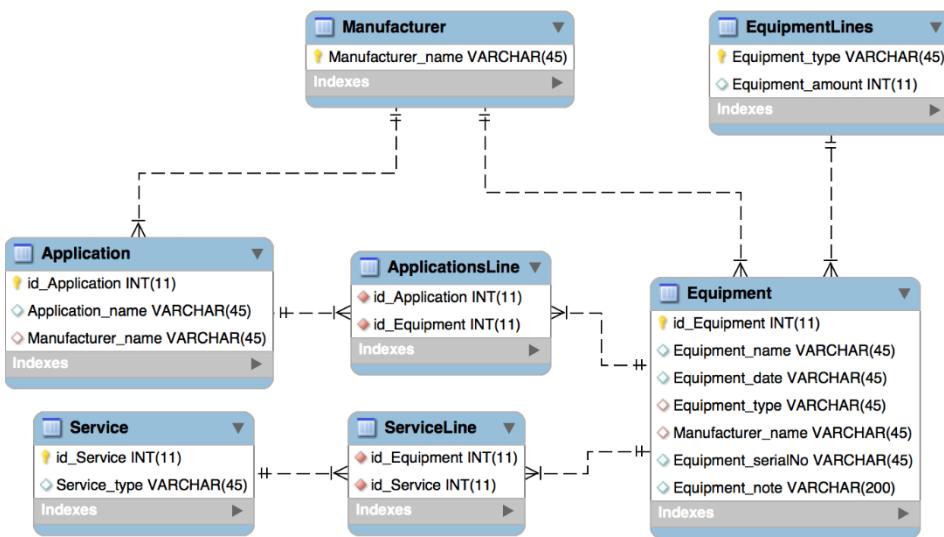
Id_App	App_name	Id_Equip	Equip_name	date	Manufacturer	type	Serial no

Dette har vi gjort for at holde bedre styr og få bedre overblik over hvilke informationer der hører til de enkelte ting.

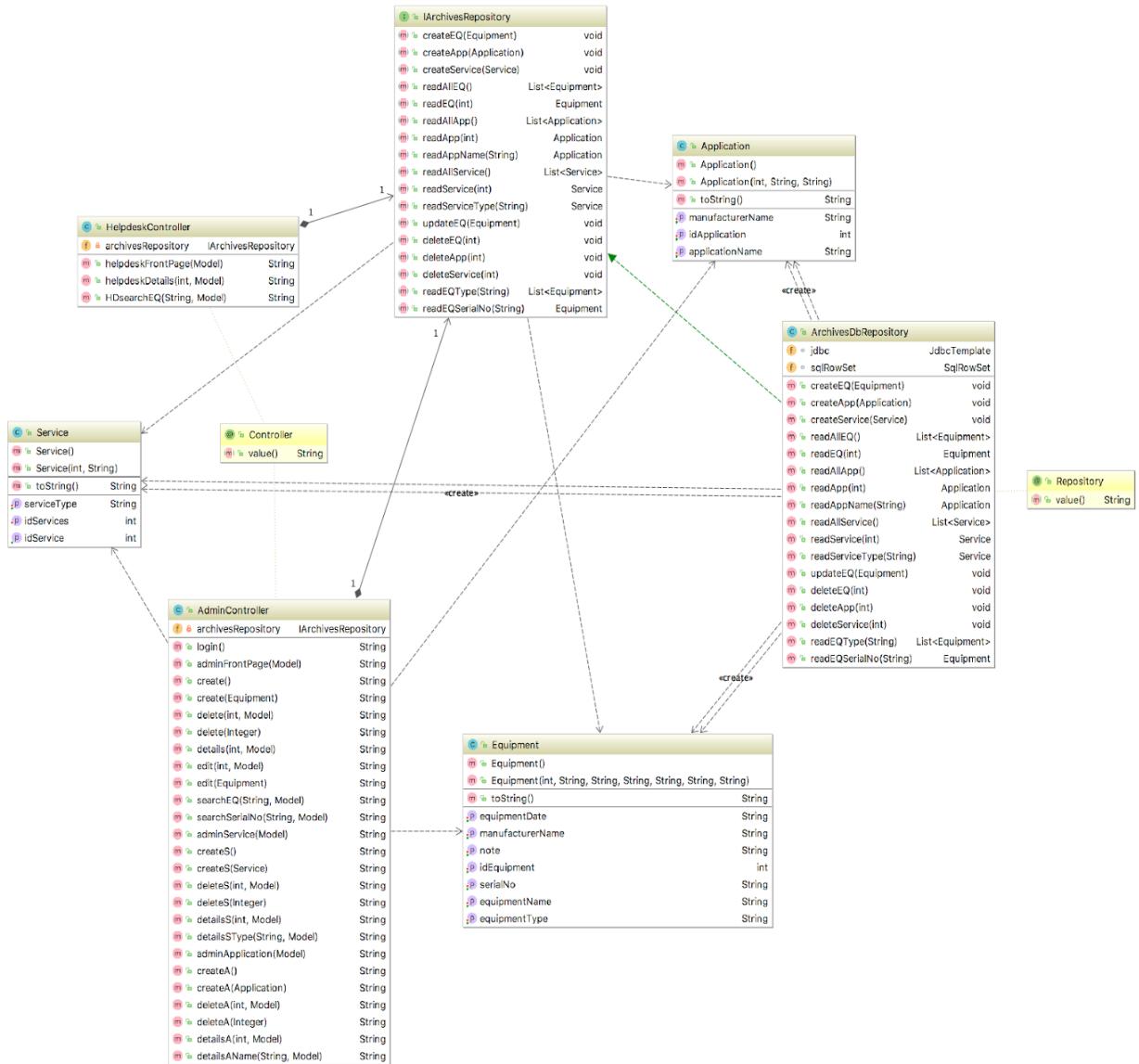
Relations skema



I denne relations tabel forekommer der "mange-til-mange" relationer ved applikation til equipment og ved service til equipment. Vi har derfor oprettet en tredje (og fjerde) tabel som kaldes for relations tabeller. Dette kan ses ved den nye vedlagte ER diagram



Her kan man se det endelig resultat af de forskellige relationer imellem Equipment, Application og Service. For at vores tabel skal hænge sammen har vi valgt at skabe tabellen applicationLine og ligeledes serviceLine. Vi valgt at refererer til de to andre tabeller ved hjælp af deres primær id-nøgle. Dette har vi gjort fordi vi vil undgå mange-til-mange relationer da der til sidste ende ville komme en masse unødvendige gentagelser.



Overgangen fra klassediagram til relations skema

Vores relation diagram er en repræsentation af det abstrakte datamodelen, hvor vores klasse diagram er en repræsentation af den statiske struktur (systemets opførelse). Mere forstående så er vores relations diagram systemets databaseperspektiv. Hvor vores klassediagram er et perspektiv på vores applikationsdomæne. De har nogle fælles objekter for at holde styr på de semantiske forhold mellem dem.

Interessante kode beskrivelser

Tilføj:

Dette er vores tilføj et nyt it-udstyr metode kaldt createEQ fra vores ArchivesDbRepository.

I denne klasse har vi lavet en skabelon til at skabe vores Equipment klasse. Dette kan ses fordi klassen skal indeholde et Equipment det hedder equipment. (**public void createEQ(Equipment equipment)**).

Vi beder om at den skal tage Equipments klasse datastruktur og indsætte (get) det indtastet information, i datastrukturens skabelon og skabe et nyt objekt med de nye værdier.

```
//Creates a new it Equipment
@Override
public void createEQ(Equipment equipment) {
    jdbc.update("INSERT INTO naviairs.Equipment " +
        "(Equipment_name, Equipment_date, Equipment_type, Manufacturer_name,
Equipment_serialNo, Equipment_note) " +
        "VALUES (" + equipment.getEquipmentName() + ", " + equipment.getEquipmentDate() + ", " +
        equipment.getEquipmentType() + ", " + equipment.getManufacturerName() + ", " +
        equipment.getSerialNo() + ", " + equipment.getNote() + ")");
}
```

Læs:

Vi har i vores program fire forskellige måder at læse et udstyr på. Overordnet set ser det meget ens ud men ”søgefunktionen” er anderledes i alle tre. I **readEQ** beder den om et int som vi har deklareret/navngivet id. Denne bruger vi til at opdater og slette udstyr, dvs. for at opdatere eller slette kalder vi **readEQ** for at finde det aktuelle id.

```
//prints one equipment with specific id
@Override
public Equipment readEQ(int id) {
    String sql = "SELECT * FROM naviairs.Equipment WHERE id_Equipment=" + id;
    sqlRowSet = jdbc.queryForRowSet(sql);

    while (sqlRowSet.next()){
        return new Equipment(sqlRowSet.getInt("id_Equipment"),
            sqlRowSet.getString("Equipment_name"),
            sqlRowSet.getString("Equipment_date"),
            sqlRowSet.getString("Equipment_type"),
            sqlRowSet.getString("Manufacturer_name"),
            sqlRowSet.getString("Equipment_serialNo"),
            sqlRowSet.getString("Equipment_note"));
    }
}
```

```

    }
    return null;
}

//bruger readEQ

@GetMapping("/aDeleteEquipment")
public String delete(@RequestParam("id") int id, Model model) {
    Equipment eq = archivesRepository.readEQ(id);
    model.addAttribute("equipment", eq);
    return "aDeleteEquipment";
}

@PostMapping("/aDeleteEquipment")
public String delete(@ModelAttribute ("id") Integer id){
    archivesRepository.deleteEQ(id);
    return "redirect:/adminEquipment";
}

@GetMapping("/aUpdateEquipment")
public String edit(@RequestParam("id") int id, Model model){
    model.addAttribute("equipment", archivesRepository.readEQ(id));
    return "aUpdateEquipment";
}

@PostMapping("/aUpdateEquipment")
public String edit(@ModelAttribute Equipment equipment){
    archivesRepository.updateEQ(equipment);

    return "redirect:/adminEquipment";
}

```

readEQSerialNo minder meget om **readEQ**. Forskellen er bare vi i **readEQSerialNo** søger på en String som er serienummeret.

```

//prints one equipment with a specific serial no.
@Override
public Equipment readEQSerialNo(String serialNo) {
    String sql = "SELECT * FROM naviairs.Equipment WHERE Equipment_serialNo LIKE '%" + serialNo +
    "%'";
    sqlRowSet = jdbc.queryForRowSet(sql);

    while (sqlRowSet.next()){
        return new Equipment(sqlRowSet.getInt("id_Equipment"),
            sqlRowSet.getString("Equipment_name"),
            sqlRowSet.getString("Equipment_date"),
            sqlRowSet.getString("Equipment_type"),
            sqlRowSet.getString("Manufacturer_name"),
            sqlRowSet.getString("Equipment_serialNo"),
            sqlRowSet.getString("Equipment_note"));
    }
}

```

```

    }

return null;
}

```

readEQType og **readAllEQ** er næsten identiske da de begge returnerer en arraylist. Forskellen på disse to er SQL koden. **readAllEQ** returnerer alle equipment skrevet ind i databasen.

readEQType sender kun dem, der indeholder en bestemt streng tilbage. Så **readEQType** bliver brugt til at få en liste ud med den søgte equipment type, hvor **readAllEQ** bliver brugt til at få hele equipment listen ud.

Formålet ved at skabe **readEQTyp** er at helpdesk skal kunne søge efter en bestemt equipment type. Hvis der kommer en der har brug for en laptop, kan helpdesk få en liste ud af alle de indregistrerede laptops.

```

//prints a list of equipment with specific equipment type
@Override
public List<Equipment> readEQType(String string) {
    List<Equipment> equipment = new ArrayList<>();
    String sql = "SELECT * FROM navairs.Equipment WHERE Equipment_type LIKE '%" + string + "%'";
    sqlRowSet = jdbc.queryForRowSet(sql);

    while (sqlRowSet.next()){
        equipment.add( new Equipment(sqlRowSet.getInt("id_Equipment"),
            sqlRowSet.getString("Equipment_name"),
            sqlRowSet.getString("Equipment_date"),
            sqlRowSet.getString("Equipment_type"),
            sqlRowSet.getString("Manufacturer_name"),
            sqlRowSet.getString("Equipment_serialNo"),
            sqlRowSet.getString("Equipment_note")));
    }

    return equipment;
}

//prints all it Equipment
@Override
public List<Equipment> readAllEQ() {
    List<Equipment> equipment = new ArrayList<>();
    String sql = "SELECT * FROM navairs.Equipment";
    sqlRowSet = jdbc.queryForRowSet(sql);
}

```

```
while (sqlRowSet.next()) {
    equipment.add( new Equipment(sqlRowSet.getInt("id_Equipment"),
        sqlRowSet.getString("Equipment_name"),
        sqlRowSet.getString("Equipment_date"),
        sqlRowSet.getString("Equipment_type"),
        sqlRowSet.getString("Manufacturer_name"),
        sqlRowSet.getString("Equipment_serialNo"),
        sqlRowSet.getString("Equipment_note")));
}

return equipment;
}
```

Vurdering af UP-metoden

Inception

I inception-fasen brugte vi tid på at indsamle en masse krav ved hjælp af interview med virksomheden, for at få et bedre billede af Naviairs behov og vision. Vi lavede en risiko analyse over projektet, hvor vi fandt de eventuelle risici der kunne opstå undervejs. Vi startede så småt med at finde de første use cases, og fik sat dem ind i en use case model. Vi fik også påbegyndt en form for klassestruktur til domænemodellen.

Elaboration

Da vi dykkede dybere ned i tingene i elaboration-fasen blev der ændret på en del ting. I vores use case model fik vi slettet nogle use cases og tilføjet *searchEquipment*. Denne use case er en ekstrem vigtig del af vores færdige system, da det selve systemet skal bruges til, er at søge i Naviairs IT-udstyrs arkiv. Vi fik lavede fully dressed på to af vores vigtige use cases (*manageEquipment*, *searchEquipment*), og fik også opdateret vores domæne model.

Vi lavede SSD diagrammer over dele af nogle use cases, for at få en bedre idé om interaktionen mellem brugeren og systemet.

Construction

Da vi gik i gang med construction-fasen, fik vi lavet SD – og klassediagram. Det blev i denne fase klart for os, at vi ikke havde fået tilrettelagt projektet godt nok fra start af. Vi havde brugt for meget tid på inception – og elaboration-fasen. Derfor valgte vi ikke at lave login-funktionen, med

brugernavn og kodeord, da vi simpelthen ikke havde tid til det. Vi nåede heller ikke at få forbundet applikationerne og ydelserne til IT-udstyret med inner joins, også grundet tidsmangel.

Konklusion

I dette eksamensprojekt skulle vi lave et arkivsystem til Naviair, da de havde brug for en måde at holde styr på alt deres it-udstyr. De ønskede en database applikation, som også kunne køres på en webbrowser. Vores mål var at skabe et website der var brugervenligt og enkelt, og som afspejlede Naviairs identitet. Det mener vi var har udført. Vi har lavet en enkel hjemmeside med et simpelt sidelayout, der både er stilren og brugervenlig. Vi har kigget på Naviairs egen hjemmeside, og har ved hjælp af farver prøvet så vidt så muligt, at skabe en flydende overgang mellem de to sider. Da Naviair er en anerkendt virksomhed, føler vi at det er vigtigt, at hjemmesiden også udstråler samme stil og klasse.

Selve projektforløbet har været fyldt med en masse forskellige udfordringer, som helt sikkert har sat hjernerne i gang, og testet vores faglige evner.

Det er første gang vi sådan rigtig har kunne arbejde ud fra UP-metoden, da de forrige projekter ikke har været lige så store som dette. Vi synes helt klart at det var en fordel at bruge UP, da det hjalp os med at få genovervejet og forbedret mange ting i udviklingsprocessen. Vi har dog erfaret at vi en anden gang, skal have mere fokus på at planlægge vores tid i starten af projektet. Vi fik brugt for lang tid på ITO - og designdelen af opgaven, og undervurderede hvor lang tid selve programmeringen ville tage. Dette resulterede desværre i at programmet ikke nåede at blive helt færdigt.

Vi synes dog alligevel selv at vi har opnået vores læringsmål, selvom det ikke rent praktisk var muligt at frembringe et helt fuldendt system, da dette ville have krævet mere tid.

Kode henvisning

Amazon Web Services: <http://naviairarchives-env.tfpriy3py9.eu-central-1.elasticbeanstalk.com/>

GitHub: <https://github.com/Veronique0165/archivesNaviair>

Litteraturliste

- *Check-in.* (28. oktober 2010).
Hentet fra <https://www.check-in.dk/naviair-er-blevet-selvstaendigt/>
- *Naviair - Aireon.* (u.d.).
Hentet fra Naviair: <http://www.naviair.dk/aireon.1238.aspx>
- *Naviair - COOPANS.* (u.d.).
Hentet fra Naviair: <http://www.naviair.dk/coopans.659.aspx>
- *Naviair - Det laver vi.* (u.d.).
Hentet fra Naviair: <http://www.naviair.dk/det-laver-naviair.1851.aspx>
- *Naviair - Mission og vision.* (u.d.).
Hentet fra Naviair: <http://www.naviair.dk/mission--vision.446.aspx>
- *Naviair - Pressemeldelse.* (2007).
Hentet fra <http://www.naviair.dk/casimo-nyhedsviser-m.-redigerbar-faktaboks.579.aspx?newsid579=819&layout=2>
- Organisation. (2012). I E. S.-H. Hans Jørgen Skriver, *Organisation* (s. 277-281). Trojka.

Københavns Erhvervsakademi
Datamatiker Uddannelsen
Systemudvikling

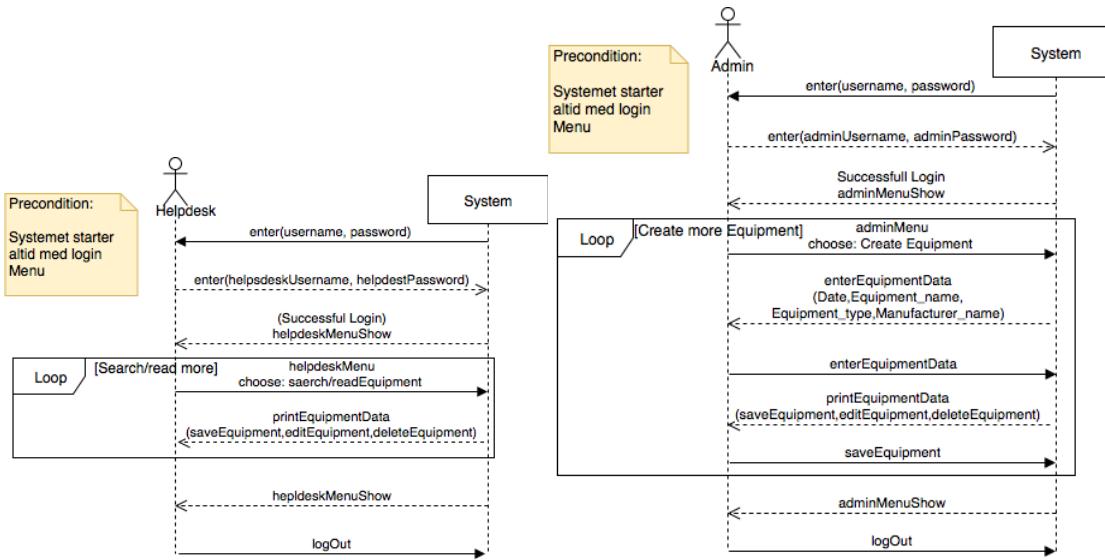


Martine Garde Magh



Veronique Cacho Basson Jensen

Bilag 1



Bilag 2: Vejledning til Naviair Archives

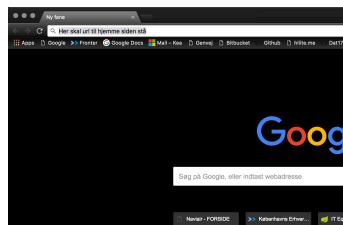
(IKKE LAVET)

a. Vejledning for at logge ind:

- Først åbner man en internetbrowser så som:



- Så taster man url'en ind:



- Hvor efter man kommer man får to følgende val muligheder
 Så vælger man om logger ind som admin eller som helpdesk.



- Man bliver så sendt videre, der skal man indtaste sit brugernavn og kode:
 Administrator Login-side Helpdesk login-side

b. Vejle

- Man logger ud ved at trykke på Logout knappen oppe i højre side.

Logout

Administrators vejledning:

c. **Vejledning for at håndtere it-udstyr.**

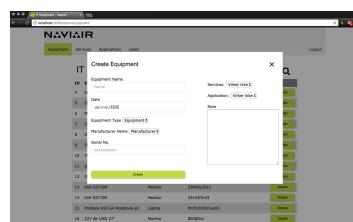
Håndtere indeholder at *tilføje, redigere, slette, søge og se* it-udstyr.

Equipment

Alt dette forgår på equipment side.

Tilføje

- Man trykke på pluset  for at tilføje et it-udstyr.



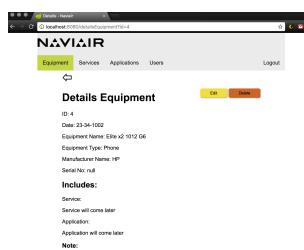
Hvor i man intaster alle udstyrets informatil den bedre om og derefter trykker create
Redigere

- Man kan redigere ved finde udstyr på 2 forskellige måde
 Et. Finde den i rækken.

Så skal man først trykker  man på details knappen:

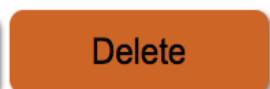
To. Finde udstyret ved at søge på det.
(Søge kan findes i punkt g. Vejledning for at se og søg it-udstyr)

Når man har gjort en af de overstående muligheder kommer man frem til udstyret detalje side.

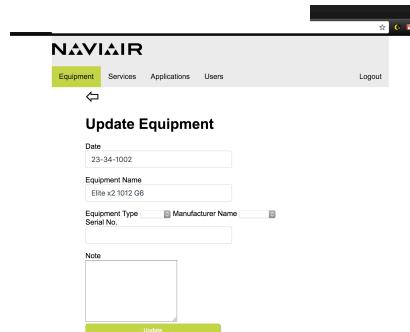


Tryk der efter på

 Edit

 Delete

Hvor de så bliver sendt til update side:



Og her kan de så ændre i de nødvendige informationer
 Husk at trykke gem når de er færdig med at lave ændringer.

Slette

3. Man kan slette ved finde udstyr på 2 forskellige måde
 Et. Finde den i rækken.

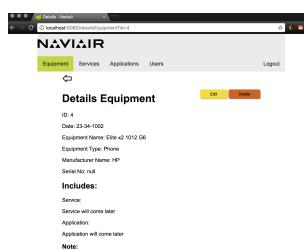
Så skal man først trykker
 man på details knappen:

Details

To. Finde udstyret ved at søge på det.

(Søge kan findes i punkt g. Vejledning for at se og søg it-udstyr)

Når man har gjort en af de overstående muligheder kommer man frem til udstyret detalje side.

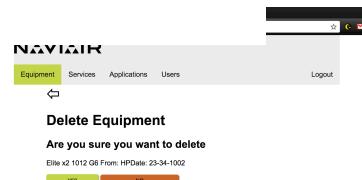


Tryk der efter på

Edit

Delete

Hvor de så bliver sendt til delete side:



Og her kan de så vælge at slette ved at trykke ja eller nej.

YES

NO

Søge og se kan findes i punkt g. Vejledning for at se og søg it-udstyr

d. Vejledning for at håndtere Service

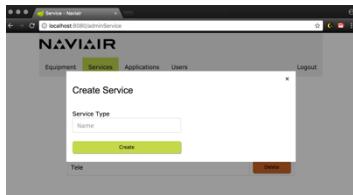
At håndtere Service indeholder at **tilføje, slette, søge og se**.

Services

Alt dette forgår på service side.

Tilføje

1. Man trykker på pluset  for at tilføje et it-udstyr.



Så popper dette vindue op:

Hvor i man indtaster alle udstyrets information den bedre om og derefter trykker create

Slette

1. Man kan slette ved finde udstyr på 2 forskellige måde

Et. Finde den i rækken.

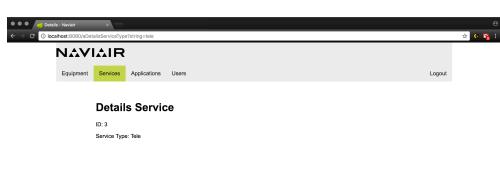
Så skal man først trykker
man på details knappen:



To. Finde udstyret ved at søge på det.

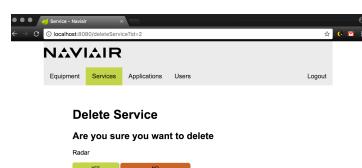
(*Søge* kan findes i punkt g. Vejledning for at se og søger it-udstyr)

Når man har gjort en af de overstående muligheder kommer man frem til udstyret detalje side.





Tryk der efter på knappen delete:



Og her kan de så vælge at slette ved at trykke endt ja eller nej.





Søge og *se* kan findes i punkt h. Vejledning for at se og søger Service

e. Vejledning for at håndtere Application

At håndtere Application indeholder at *tilføje, slette, søge og se*.

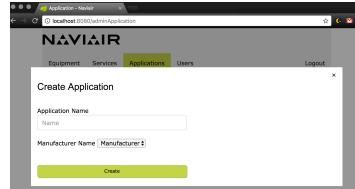
Applications

Alt dette forgår på application side.

Tilføje

1. Man trykker på pluset  for at tilføje et applikation.

Så popper dette vindue op:



Hvor i man indtaster alle applikationens information den bedre om og derefter trykker create

Slette

2. Man kan slette ved finde udstyr på 2 forskellige måde
 Et. Finde den i rækken.

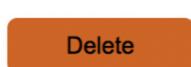
Så skal man først trykker  man på details knappen:

To. Finde udstyret ved at søge på det.

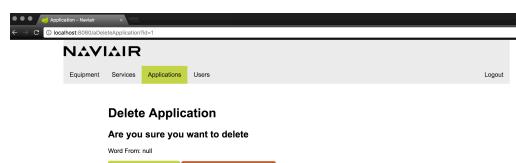
(*Søge* kan findes i punkt g. Vejledning for at se og søg it-udstyr)

Når man har gjort en af de overstående muligheder kommer man frem til udstyret detalje side.



 Delete

Tryk der efter på knappen delete:



Og her kan de så vælge at slette ved at trykke endt ja eller nej.

 YES

 NO

Søge og se kan findes i punkt i. Vejledning for at se og søg Application

- f. Vejledning for at håndtere User
Håndtere indeholder at *tilføje, redigere, slette, søge og se* User.
(IKKE LAVET)

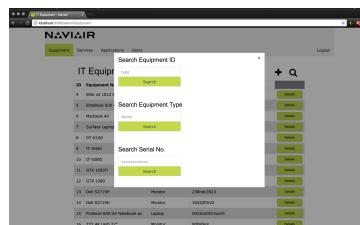
Helpdesk vejledning:

- g. Vejledning for at se og søger it-udtysr

Så skal man først trykke på søge tegnet:



på Equipment siden:



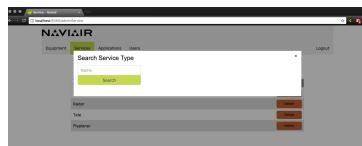
Hvor dette vindue popper op:
her skal man indtaster det udstyrs id man ledere efter

- h. Vejledning for at se og søger Service

Så skal man først trykke på søge tegnet:



på Service siden:



Hvor dette vindue popper op:
her skal man indtaster typen på servicen man ledere efter

- i. Vejledninger for at se og søger Application

Så skal man først trykke på søge tegnet:



på Application siden:



Hvor dette vindue popper op:
her skal man indtaster navnet på

