SEMESTEROPGAVE: SKAKKLUBBEN-K41

Logo hentet fra skakklubbenk41.dk

KLASSE: DAT18C

GRUPPE: 1

KØBENHAVNS ERHVERVSAKADEMI

AFLEVERINGSDATO: 07.12.2018

VEJLEDERE: ASGER, DAVID & JARL

ANTAL RAPPORT SIDER: 31

UDARBEJDET AF DÉSIRÉE, SOFIE, PÄIVI & ANDREAS

Indholdsfortegnelse

Indledning/Virksomhedsbeskrivelse (Tekst af Désirée)

Skakklubben K41 er en mindre skakklub, der i den seneste tid har oplevet op- og nedgangstider. Derfor har ledelsen valgt, at de gerne vil udvikle et administrativt system, der skal styre medlemsoplysninger, kontingenter og spillernes styrke. Vores opgave er at udarbejde forretningsgrundlag (mission og vision) og lave en SWOT-analyse for Skakklubben K41. Vi har også valgt at lave en risikoanalyse til projektet og en interessentanalyse med udgangspunkt i skakklubben. Vi har også et Entity Relationship diagram med som viser vores opbygning af vores filer som vi bruger i vores system. Vores projekt bliver udviklet som en UP-proces med hertil knyttede UML-analyseværktøjer:

- Use cases
- Use case diagram
- Domain model
- SSD
- SD
- Class diagram

Gruppen har ikke valgt udpege nogen projektleder, men i fællesskab styre projektet.
Vi har valgt at bruge Trello-programmet som Kanban-værktøj som skal hjælpe os med at beholde overblikket med hvor vi er henne i projektet og hvad der mangler at blive lavet.

Forretningsgrundlag (Tekst af Désirée)

Mission

Vores skakklub er andet end bare en skakklub. Vores mission er at skabe et fællesskab for vores medlemmer udover at hygge sig med at spille skak. Skakklubben K41 er med til at bidrage til at skabe større interesse i et godt brætspil. Vi vil medvirke til at stimulere og udvikle hjernen hos unge såvel som ældre.

Business vision

Skakklubben K41's vision er at løbende udvikle nogle af Nordens bedste skakspillere, som skal være med i eliten. Vi vil stræbe efter at vinde et Danmarksmesterskab senest i 2023. Skakklubben ønsker sig en mere stabil udvikling i deres medlemstal samtidig med at medlemstallet skal stige mindst 10% de næste fem år. For at Skakklubben skal være mere professionel har vi valgt at investere i et nyt it-system, som skal være med til at få et bedre overblik over vores medlemskaber. Udover det ønsker vi at tilbyde sjove tilbud til alle niveauer.

Afgrænsning af opgaven (Tekst af Désirée)

Vi har valgt at kigge på skakklubben og vil udarbejde en prototype af et administrationssystem der holder styr på deres medlemmer samt deres betaling som vi mener der vil passe til deres ønsker. På grund af tidsmangel har vi valgt ikke at implementere ratings i vores prototype, og vores system har derfor ikke nogen kontakt med ELO-systemet. Vores prototype tager altså kun udgangspunkt i at etablere medlemsdelen til skakklubben da vi mener at det er kernen i det administrativ system.

Krav (Tekst af Sofie)

Kravene til vores administrationsprogram stammer fra opgavebeskrivelsen, og vi har selvsagt ikke været løbende i kontakt med den fiktive kunde. Disse krav er også skrevet på baggrund af afgrænsningen af opgaven i foregående afsnit og vil derfor ikke indeholde ratingdelen.

Skakklubben K41 vil have udviklet et administrationssystem til at styre deres medlemmer, og derfor skal systemet kunne håndtere alle de klassiske CRUD-elementer:

- C: Oprette nye medlemmer.
- C/R: Opdele medlemmer i forskellige kontingentkategorier.
- R: Se hvilke medlemmer, der er i restance.
- R: Se hvilke medlemmer, der er junior.
- R: Søge oplysninger om et bestemt medlem.
- U: Opdatere medlemmernes oplysninger.
- D: Slette et medlem fra klubben.

For at imødegå det har vi besluttet at lave en database med filer over medlemmerne. Selve programmet bliver menustyret.

De forskellige kontingenttyper er som følger:

Aktiv:

- Junior (under 18 år): 200 kr. årligt.
- Senior: 600 kr. årligt.
- Senior60+ (over 60 år): 25 % rabat af seniorprisen.

Passiv:

- 100 kr. årligt.

Er medlemmet under 18 år, skal CPR-nummeret oplyses ved indmeldelse, da klubben kan få tilskud fra kommunen, hvis klubben sender en liste med juniormedlemmer til kommunen. Opbevares CPR-numre i systemet, skal disse krypteres.

Da vi som sagt ikke har været i kontakt med vores fiktive kunde, er vi på egen hånd blevet enige om følgende vedrørende klubbens styring af medlemmerne og betaling:

- Der betales kontingent én gang om året i anden halvdel af januar, hvor hele året betales.
- Kassereren indberetter juniormedlemmerne til kommunen i første halvdel af januar, hvorefter medlemmerne rykker kontingent, hvis nogen er gået fra junior til senior eller fra senior til senior60+.
- Kontingentet opkræves via et eksternt faktureringssystem, d.v.s. at vores program ikke sender regninger eller påmindelser ud til medlemmerne.
- Har man ikke betalt sit kontingent efter to måneder, ryger man på den sorte liste og meldes ud af klubben. Forsøger et medlem at melde sig ind igen, kan personen ikke dette, hvis personen står på den sorte liste.
- Når man melder sig ind i klubben, har man ligeledes to måneder til at betale sit kontingent.

Afgrænsning af krav:
Da vores program skal ses som en prototype jævnfør afgrænsningen af opgaven, krypterer vi ikke medlemmernes CPR-numre. Alle personer og derfor også alle CPR-numre er fiktive i denne prototype. CPR-numrene er taget med for at vise funktionerne af de dele af programmet, som vedrører juniorerne.

FURPS+ (Tekst af Andreas)

For at udarbejde et system der er fyldestgørende, er stikordene FURPS+ en måde at sikre sig, at man får gjort de vigtige overvejelser, om hvad systemet skal kunne og indeholde.

Functionality	Programmet skal præsentere en konsolmenu som brugergrænseflade, der giver brugeren adgang til programmets features. Systemet gemmer vigtige oplysninger om medlemmer og betalinger i filer som ligger lokalt på computeren. Filerne er struktureret som tabeller ville optræde i en relationel database.
Usability	Konsolmenuen har også til formål løbende at forklare brugeren, hvordan programmet fungerer, og at indlæse input fra brugeren, hvor der er behov for det.
Reliability	Backup findes online på github.
Performance	Database filer skal være indekserede, hvis klubben opnår et meget højt antal medlemmer. Dette ville mindske runtime for at udføre skrivning eller sletning inde i filerne. (Ikke implementeret)
Supportability	Programmet laves som en webløsning med login fra administrator. Lige nu understøttes det kun af java.
+	Programmet understøtter i øjeblikket kun dansk sprog. Det er et lovkrav at kryptere CPR-numre.

Ordliste (Tekst af Sofie)

Aktør	Brugeren af systemet. Enten formanden eller kassereren af skakklubben.
CPR (fil)	CPR er navnet på en fil, der indeholder en liste med CPR-numre på juniormedlemmer.
ELO-system	Det danske ratingsystem for skakspillere. Ratingsystemet er ikke implementeret her, og derfor har medlemmer ikke ratings i denne prototype.
erTurneringsspiller	Man er enten turneringsspiller eller hyggespiller. Er man ikke registeret som turneringsspiller, er man altså hyggespiller. Turneringsspillere burde få ratings, når de oprettes, men dette sker ikke i vores prototype grundet afgrænsning af opgaven.
Filhaandtering	En statisk klasse, der tager sig af håndteringen af filer; det vil sige læsning af og skrivning til filer.
at gemme	Når vi gemmer i vores program, skriver vi til en fil.
Junior	Medlemmer under 18 år.
Juniorlisten	Liste af medlemmer, der er under 18 år. Indeholder CPR-nummer, navn og adresse. Listen skal sendes til kommunen for at få tilskud.
Kontingenttyper (fil)	Kontingenttyper er navnet på en fil, der indeholder en liste med kontingenttyper. De forskellige kontingenttyper er junior, senior og senior60+ og passive medlemmer.
Kunden	Skakklubben K41.
Medlemmer (fil)	Medlemmer er navnet på en fil, der indeholder en liste med alle medlemmer. Medlemmer refererer også til individer i klubben.
Medlemsform	De forskellige medlemsformer er aktiv/passiv og turnerings-/hyggespiller.
Senior	Medlemmer mellem 18-60 år.
Senior60+	Medlemmer over 60 år.
Skakklubben K41	Også kaldet: klubben, skakklubben, kunden.
at slette medlemmer	Da dette er en prototype med fiktive medlemmer, overholder programmet ikke nødvendigvis GDPR. Et slettet medlem vil få medlemsnummeret 00.
den Sorte Liste (fil)	Liste af tidligere medlemmer, der ikke har betalt. Man kommer på den sorte liste og bliver udmeldt klubben, hvis man har været i restance i 2 måneder. Det tjekkes for alle nye medlemmer, om de er på den sorte liste, da man ikke kan melde sig ind igen, hvis man står på listen. Den sorte liste er også navnet på en fil.

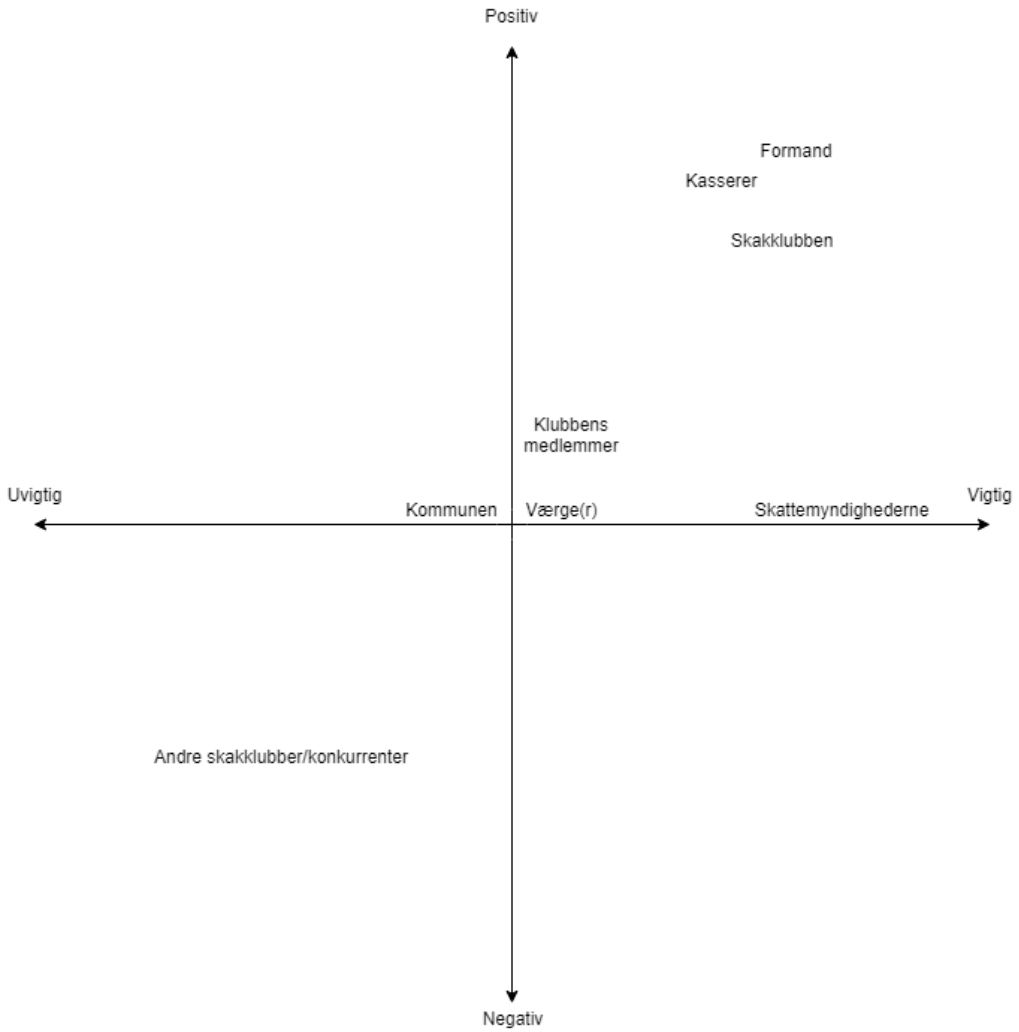
Interessent-analyse (Tekst af Andreas)

Interessentanalysen er udarbejdet i forhold til klubben og ikke i forhold til projektet.

Identificering af interessenter:

Formand: Ønsker at holde styr på (nye) medlemmer i klubben.
Kasserer: Ønsker overblik over indbetaling af kontingenter, eller mangel på samme, af samtlige medlemmer i klubben.
Andre skakklubber/konkurrenter: Vil gerne kende til klubbens popularitet og styrke i forhold til fremtidige turneringer.
Klubbens medlemmer: Systemet sikrer en lettere adgang til kontingent og medlemsoplysninger som gavner medlemmerne.
Kommunen: Har interesse i at få oplyst antal af medlemmer under 18 år og udbetale tilskud.
Skakklubben: Ønsker flere/nye medlemmer og bedre overblik over nuværende medlemmer og deres kontingentbetalinger.
Skattemyndigheder: Opkrævning af skatter og moms m.m.
Værge(r) for juniorspillere: Skal have den rigtige kontingenttakst.

Klarlæggelse af interessenternes indflydelse



Interessenternes succeskriterier

Formand: Systemet skal være funktionelt og give et hurtigt og nemt overblik over klubbens eksisterende og nye medlemmer.
Kasserer: Det bliver lettere at holde styr på kontingenttakster og medlemmers indbetalinger.
Andre skakklubber/konkurrenter: At systemet giver overblik over klubbens styrker og svagheder.
Klubbens medlemmer: Lettere tilgang gennem klubben til medlemskabshåndtering.
Kommunen: De kan sende tilskud for antallet af juniorspillere.
Skakklubben: At de kan oprette nye medlemmer og håndtere betaling af medlemskab.
Skattemyndigheder: Der bliver betalt den retmæssige og korrekte skat og moms til myndighederne.
Værge(r) for juniorspillere: At klubben får tilskud, og at de får rabat i deres kontingenttakst.

SWOT-analyse (Tekst af Sofie)

SWOT-analysen tager udgangspunkt i Skakklubben K41 som virksomhed. Den er udarbejdet på baggrund af klubbens hjemmeside inklusive de tre referater fra 2017, der er tilgængelige på hjemmesiden. Visse elementer i analysen er opfundet af os for opgavens skyld.

Styrker

- Etiske retningslinjer
- At stå ved sine etiske retningslinjer
- Fokus på juniorer

Svagheder

- Hjemmesiden: mangelfuld, rodet og ikke opdateret
- Dalende medlemstal
- Nedslidte lokaler

Muligheder

- At kunne være passivt medlem ved indmeldelse
- Reklame i lokalaviser
- Salg af mad/drikke ved turneringer
- Samarbejde med folkeskolerne
- Nyt administrationssystem

Trusler

- Tidligere medlem, der er meget offensiv og kritisk
- Underskud
- Andre klubber overtager medlemmer

Det er en styrke, at Skakklubben har vedtaget etiske retningslinjer for klubben, da det viser klubben som ansvarlig. Udsmidningen af et medlem antyder, at retningslinjerne også allerede er blevet brugt. Det viser derfor klubben fra en positiv side, når de etiske retningslinjer ikke blot er tomme ord.

Det ses tydeligt på hjemmesiden, at klubben fokuserer på juniorerne, hvilket er en styrke, da unge medlemmer har potentiale til at videreføre klubben i en ny generation. Hjemmesiden i sig selv er dog en svaghed for klubben, da den er både mangelfuld, rodet og ikke opdateret.

Det dalende medlemstal tyder på en svaghed i klubben, og det burde undersøges, hvorfor folk melder sig ud. De nedslidte lokaler har måske en betydning på det punkt. En anden årsag kunne være et tidligere medlem, der er en trussel for klubben, idet medlemmet taler kritisk om klubben, og selve årsagen til udsmidningen sandsynligvis bunder i et brud på de etiske retningslinjer. På grund af det dalende medlemstal er der en reel trussel om underskud. Truslen bliver ikke mindre af, at medlemmerne pga. det tidligere medlem muligvis forlader klubben til fordel for andre klubber.

Der er visse muligheder at tage fat på for at få flere medlemmer og mere kapital i klubben. Klubben kan med fordel reklamere i lokalaviser og samarbejde mere med skolerne og fritidsordningerne. Klubben kan også have salg af mad og drikke ved turneringer. Det vil også være en mulighed, hvis man kunne melde sig ind i klubben som passiv medlem uden først at have været aktiv, hvis man blot ønsker at støtte klubben eller eventuelt modtage et medlemsblad. Et nyt administrationssystem vil også kunne holde bedre styr på medlemmer, der ikke har betalt.

Risikoanalyse (Tekst af Päivi)

Denne risikoanalyse er udarbejdet sådan, som man ville forvente at gøre det i en rigtig situation med en kunde, som har bestilt et administrativt system. Dermed indeholder analysen nogle fiktive elementer.

Ordforklaringer:

Sandsynlighed S = hvor sandsynligt det er, at risikomomentet indtræffer på en skala fra 1 til 5:

1. Meget lav
2. Lav
3. Moderat
4. Høj
5. Meget høj

Konsekvens K = hvor alvorlige konsekvenser det har for projektet, hvis risikomomentet indtræffer på en skala fra 1 til 10:

1. Ubetydelig
3. Tålelig
7. Alvorlig
10. Katastrofal

Risikotal: $R = S \cdot K$

Aktionsplan: hvordan risikomomentet undgås, og hvad der skal gøres, hvis det alligevel indtræffer. Der rettes særlig opmærksomhed på alle risikomomenter med risikotal på over 9 i overensstemmelse med aktionsplanen. Der foretages løbende risikovurdering, og aktionsplanen justeres efter behov.

Risikoanalyse med aktionsplan

Risikomoment	S	K	R	Aktionsplan
Projektrisici				
1. Projektgruppen kan ikke holde tidsplanen og opnår ikke den næste planlagte milepæle	1	3	3	Gruppen styrer tidsplanen vha. Kanban. Hvis tidsplanen skrider, laves der en prioriteringsliste over de mest presserende opgaver samt tidsplanen justeres, således at gruppen kan nå de aftalte milepæle.
2. Projektgruppen mangler kompetencer, der er vigtige ift. opgaveløsningen.	1	7	7	Gruppen beder om hjælp fra undervisere og mere erfarne medstuderende.
3. Der opstår problemer eller misforståelser i forbindelse med kommunikationen med kunden.	2	7	14	Gruppen informerer kunden løbende om projektets fremgang. Hvis kommunikationen går helt i stå med kunden, bedes bestyrelsesformanden i skakklubben om at agere som mægler.
4. Skakklubben kan ikke levere de aftalte økonomiske ressourcer til projektet.	1	10	10	Skakklubben har allerede fået allokeret ressourcer til projektet fra Den Danske Skakunion. Skakklubbens erhvervsansvarsforsikring dækker tab i forbindelse med allerede udført arbejde, hvis klubben går konkurs.

Risikomoment	S	K	R	Aktionsplan
Personalerisici				
1. Nøglepersoner i skakklubben (formand / kasserer) bliver syg eller fratræder.	2	1	2	Skakklubben udpeger en anden kontaktperson.
2. Et eller flere medlemmer i projektgruppen bliver syge eller af andre personlige grunde ikke kan deltage i projektet.	3	3	9	Medlemmer prøver at undgå smittekilder. Stress forebygges med god planlægning. Bliver et medlem syg, kompenseres det ved at andre arbejder mere.
3. Et gruppemedlem stopper på uddannelsen under forløbet.	1	3	3	Dette vurderes ikke at danne et reelt risikomoment, idet forløbet kun varer få uger.
Organisatorisk risici				
1. Skakklubben kommer i finansielle vanskeligheder pga. kontingenter i restance.	3	1	3	Det nærværende projekt er finansieret af Den Danske Skakunion, så kontingenter har ikke direkte indflydelse på projektet. Administrationssystemet kan i fremtiden bruges til at inddrive kontingenter i restance.
2. Skakklubben har dalende medlemstal og lukkes.	1	10	10	Projektet varer kun få uger, så en eventuel lukning af klubben ville ikke kunne nå forekomme under forløbet. Se også det forudgående punkt i nærværende aktionsplan vedr. finansiering.
Produktrisici				
1. Slutbrugerne (primært formand / kasserer) kan ikke betjene systemet.	4	7	28	Gruppen giver instrukser både mundtligt og skriftligt til betjening af systemet. Kunden må gerne kontakte gruppen de følgende seks måneder i tilfælde af spørgsmål.
2. Administrationssystemet bliver udsat for et hackerangreb i løbet af projektet.	1	10	10	Gruppen sørger hele tiden for at have backup af systemet.

Risikomoment	S	K	R	Aktionsplan
Teknologisk risici				
1. Skakklubbens IT-system er ikke opdateret og kan ikke bruges til at betjene administrationssystemet.	4	7	28	Gruppen afstemmer de IT-mæssige krav til systemet med kunden og udvikler et system, som ikke stiller for store krav til software/hardware.
2. Skakklubben har ikke ressourcer til at vedligeholde systemet og/eller på sigt leve op kravene til software/hardware.	4	1	4	Dette punkt er uden for projektets rækkevidde og dermed ikke gruppens ansvar, men kunden har mulighed for at tegne et vedligeholdelsesaftale med gruppen. I det tilfælde tegnes der en ny kontrakt.

Konklusion på risikoanalysen
En af gruppemedlemmer blev småsyg i løbet af projektet, men det forårsagede ikke nogen forsinkelse takket være det pågældende gruppemedlems aktive indsats på trods af sygdommen. Alle risikomomenter blev løbende overvåget, men ingen af de andre risikomomenter indtraf.

Use cases (Hele gruppen)

Oversigt over use cases:
UC1: Oprette et nyt medlem
UC2: Fakturering af medlemmer
UC3: Se hvilke medlemmer der ikke har betalt
UC4: Rapportere liste med juniormedlemmer
UC5: Opdatere medlemmernes stamoplysninger
UC6: Opdatere medlemskab
UC7: Årlig opdatering af oplysninger
UC8: Slette medlemmer
UC9: Søge et medlem

Afgrænsning af use cases:
Vi finder det ikke realistisk tidsmæssigt at implementere alle use cases og vælger derfor at lave endnu en afgrænsning. I afgrænsningen er der lagt vægt på at få flest mulige CRUD-elementer med i vores egentlige implementering. Derfor er vi nået frem til at implementere følgende use cases:

UC1: Oprette et nyt medlem
UC3: Se hvilke medlemmer der ikke har betalt
UC5: Opdatere medlemmernes stamoplysninger
UC6: Opdatere medlemskab

Alle use cases vil dog blive beskrevet som minimum brief nedenfor, blive vist i et use case-diagram samt blive vist som konceptuelle klasser i domænemodellen.

UC1: Fully dressed (Tekst af Päivi)

Use Case Number	1 (fully dressed)
Use Case Name	Oprette et nyt medlem
Scope	Administrationssystem
Level	Brugerniveau
Primary actor	Formand
Stakeholders and interests	*Formand* – vil oprette et nyt medlem nemt med tydelige menuer og instrukser fra systemet undervejs. *Kasserer* – oprettelse af medlemmer er en forudsætning for, at kassereren kan håndtere kontingenter (se UC3, UC4 og UC7). *Skakklubben* – vil registrere medlemmer for at drive klubben (for at sende information til medlemmer, opkræve kontingenter osv.) *Medlemmer* – vil oprettes for at kunne blive medlem af klubben. *Københavns Kommune* – oprettelse af medlemmer er en forudsætning for at kommunen kan modtage oplysninger om juniorer (se UC4). *Programmører (gruppe1)* – vil lave et administrationssystem, hvor oprettelsen af nye medlemmer er en central funktion, med henblik på fremover at kunne bruge systemet i en tilsvarende opgave. *Opdragsgivere (Asger og Jarl)* – vil kunne se, at gruppen har lært denne del af pensum.
Preconditions	Aktøren har startet systemet og valgt oprettelse af nye medlemmer i menuen.
Success Guarantee	Et nyt medlem er blevet oprettet med et unikt medlemsnummer
Main Success Scenario	1. Systemet genererer et nyt medlemsnummer og en oprettelsesdato. 2. Systemet beder om fornavn, efternavn og fødselsdato. 3. Aktør indtaster fornavn, efternavn og fødselsdato. 4. Systemet tjekker, om personen er under 18. a) Hvis ja: systemet beder om cpr-nummer. Aktøren indtaster cpr-nummer. Medlemsnummeret og cpr-nummeret tilføjes automatisk til cpr-filen. b) Hvis nej: systemet beder ikke om cpr-nummeret. 5. Systemet beder aktøren om at vælge medlemsform (aktiv/passiv, hygge-/turneringsspiller). 6. Aktøren vælger medlemsform. 7. Systemet beder om telefonnummer, gadenavn, husnummer, postnummer, bynavn og email.

	8. Aktøren indtaster telefonnummer, gadenavn, husnummer, postnummer, bynavn og email. 9. Systemet bekræfter oprettelsen.
Extensions	*a. Når som helst aktøren indtaster oplysninger i en forkert form (bogstav i stedet for tal, postnummeret med for få cifre og lignende): 1. System beder aktøren om at indtaste oplysningen i den ønskede form *b. Når som helst oprettelsen bliver afbrudt 1. Medlemmet bliver først gemt, når alle oplysninger er indtastet, så aktøren er nødt til at lave en ny oprettelse. *c. Når som helst aktøren indtaster forkerte oplysninger 1. Medlemmet bliver først gemt, når alle oplysninger er indtastet. Når medlemmet er blevet oprettet, kan oplysninger ændres (se UC5). 3c) Aktøren indtaster oplysninger i søgefeltet forkert, så systemet finder ikke medlemmet og opretter et nyt, selv om medlemmet eksisterer i databasen. 1. Efter faktureringsoplysninger er blevet sendt til det eksterne faktureringssystem, får kassereren besked fra faktureringssystemet om, at det pågældende medlem allerede har betalt (se UC2). Kassereren sletter den seneste oprettelse (se UC8). 8a) Medlemmet mangler en oplysning, fx hvis personen ikke har et telefonnummer 1. Aktøren indtaster \N i det pågældende felt. 9a) Medlemmet findes i forvejen. 1. Systemet giver besked at medlemmet allerede findes i systemet. Oprettelsen afbrydes. 9b) Medlemmet eksisterer ikke i systemet, men er på den sorte liste. Systemet giver besked om at medlemmet er på den sorte liste. Oprettelsen afbrydes.
Frequency of Occurrence	Kontinuerlig

UC2 (Tekst af Sofie)

Fakturering af medlemmer
Precondition: Alle medlemmer har en kontingenttype. Fakturering af eksisterende medlemmer sker først efter UC7.
Main success scenario: Fakturaoplysningerne sendes automatisk til et eksternt faktureringssystem, der sender en faktura til medlemmerne. Fakturaen sendes hver januar den 15. og indeholder forfaldsdatoen 31. januar. I tilfælde af et nyt medlem sendes fakturaen den 15. i følgende måned med forfaldsdato den sidste dag i den måned. Hvis et medlem fejlagtigt er blevet oprettet to gange i systemet pga. fejl i forbindelse med oprettelsen (se UC1), giver faktureringssystemet kasserer besked om dette. Kasserer kan slette den seneste oprettelse (se UC8).

UC3 (Tekst af Andreas)

Se hvilke medlemmer der ikke har betalt
Precondition: UC2.
Main success scenario: Aktøren ønsker at vide, hvilke medlemmer der ikke har betalt, og får programmet til at skrive en liste i en ny fil, da denne kan vise sig at være stor/indeholde mange oplysninger. Følgende info udskrives til filen: medlemsnummer, fornavn og efternavn og hvad medlemmet skylder.

UC4 (Tekst af Sofie)

Rapportere liste med juniormedlemmer
Precondition: I forbindelse med oprettelse af et nyt medlem er medlemsnummer og CPR-nummer blevet tilføjet til CPR-filen, når medlemmet er under 18 år (se UC1).
Main success scenario: Aktøren udskriver en liste (med CPR-nummer, navn og adresse) med antal juniormedlemmer. Denne liste sendes til kommunen for at få tilskud.

UC5 (Tekst af Désirée)

Opdatere medlemmernes stamoplysninger
Main success scenario: Et medlem ønsker at ændre sine stamoplysninger. Aktøren indtaster medlemsnummeret i systemet på det medlem, der skal have opdateret sine stamoplysninger. Aktøren laver herefter de ændringer det pågældende medlem har. Systemet gemmer de nye opdateringer.

UC6 (Tekst af Désirée)

Opdatere medlemskab
Main success scenario: Et medlem ønsker at ændre sin medlemsstatus fra aktiv/passiv eller hygge-/turneringsspiller. Aktøren indtaster medlemsnummeret i systemet på det medlem der skal have opdateret sin medlemsstatus. Aktøren laver den nødvendige opdatering. Systemet gemmer de nye oplysninger.

UC7 (Tekst af Andreas)

Årlig opdatering af oplysninger
Precondition: UC4.
Main success scenario: Aktøren navigerer i menuen hen til den årlige opdateringsfunktion. Systemet udfører herefter opdatering, hvor alle står som værende ikke betalt for det nye år, og systemet sletter automatisk CPR-numre på de medlemmer, der ikke er juniorer længere. Systemet udregner nu automatisk på baggrund af fødselsdato, om et medlem har fået en ny kontingenttype.

UC8 (Tekst af Andreas)

Slette medlemmer
Main success scenario: Et medlem ønsker at blive slettet som medlem af klubben. Aktøren navigerer ind til slettefunktionen i menuen og indtaster medlemsnummer. Systemet sletter herefter alle oplysninger i systemet med det pågældende medlemsnummer, såfremt der ikke mangler at blive betalt medlemskontingent. Hvis ikke medlemskabskontingentet er betalt, bliver brugeren ikke slettet, men i stedet gør systemet kasserer/formand opmærksom på dette. Skulle medlemmet ønske at blive slettet uden at have mangle at betale, markeres medlemmet i filen som slettet, uden at resten af medlemmets oplysninger bliver fjernet helt.

UC9 (Tekst af Sofie)

Søge et medlem
Main success scenario: Aktøren ønsker oplysninger på et bestemt medlem. Aktøren vælger om vedkommende vil søge på et navn eller et medlemsnummer. Systemet giver alle oplysninger om det pågældende medlem.

Use Case Diagram (Tekst af Päivi)

Use Case Diagram visualiserer use cases og viser, hvordan de relaterer til hinanden. Dermed afspejler diagrammet funktionerne i systemet og deres indbyrdes relationer, samt hvilken primær aktør er knyttet til hvilke use cases. Vi har to primære aktører, nemlig formand og kasserer, som betjener systemet. På højre side af systemet finder man tre sekundære aktører: filerne (jf. ordliste), det eksterne faktureringssystem og kommunen. Systemet har forbindelse til disse sekundære aktører, men de betjener ikke systemet.

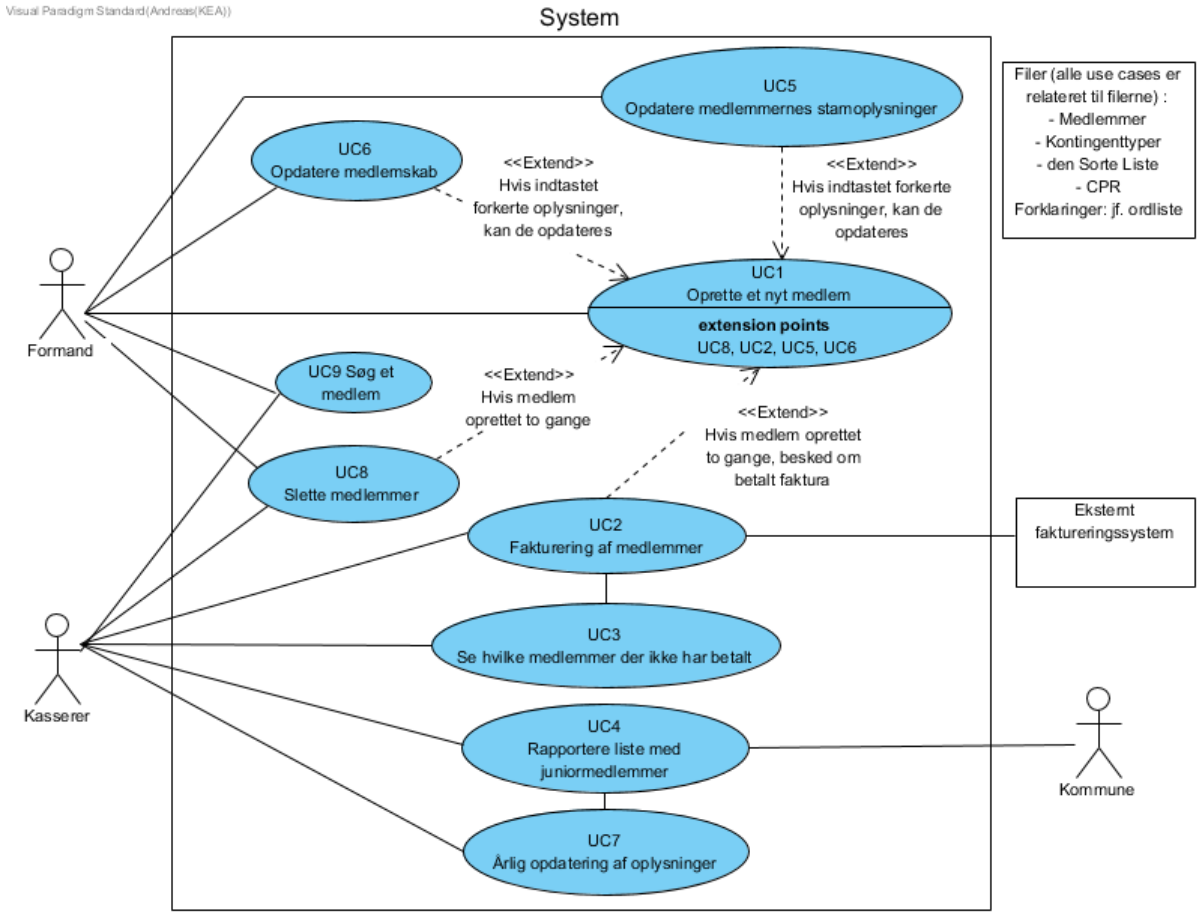
UC1 “oprette et nyt medlem” har extensions knyttet til følgende use cases: UC8, UC2 og UC5. Hvis aktøren i forbindelse med oprettelsen indtaster navnet og telefonnummeret forkert, så systemet ikke finder medlemmet og opretter et nyt, selv om medlemmet eksisterer i databasen, får aktøren besked fra faktureringssystemet om dette.Aktøren sletter den seneste oprettelse (UC8). Hvis aktøren indtaster forkerte oplysninger under oprettelsen, kan de rettes efterfølgende (UC5 og UC6).

UC2 “Fakturering af medlemmer” er associeret med UC3 “Se, hvilke medlemmer der ikke har betalt”, fordi listen over medlemmer i restance hænger sammen med fakturering af medlemmer, selv om funktionerne ikke behøver at blive udført samtidigt.

UC4 “Rapportere lister med juniormedlemmer” er ligeledes associeret med UC7 “Årlig opdatering af oplysninger”. I vores prototype har vi ikke implementeret disse to use cases, men kunden har mulighed for at få udvidet systemet. Efter juniormedlemmerne er blevet rapporteret til kommunen i januar måned, vil alle medlemsoplysninger blive opdateret således, at systemet sletter cpr-numrene for de medlemmer, der ikke er juniorer længere og medlemmerne skifter automatisk kontingenttype.

Use Case Diagrammet afspejler, på hvilken måde systemet indeholder de såkaldte CRUD-funktioner: create, read, update and delete. Vi har bl.a. mulighed for at oprette et medlem, læse, hvilke medlemmer har betalt og opdatere et medlem.

Use Case Diagram (Hele gruppen)



Domain Model (Tekst af Päivi)

Domænemodellen viser, hvilke oplysninger vi har brug for i vores system og hvad relationerne mellem disse oplysninger er. Domænemodellen dækker samtlige use cases i vores system illustreret af konceptuelle klasser og deres indbyrdes relationer.

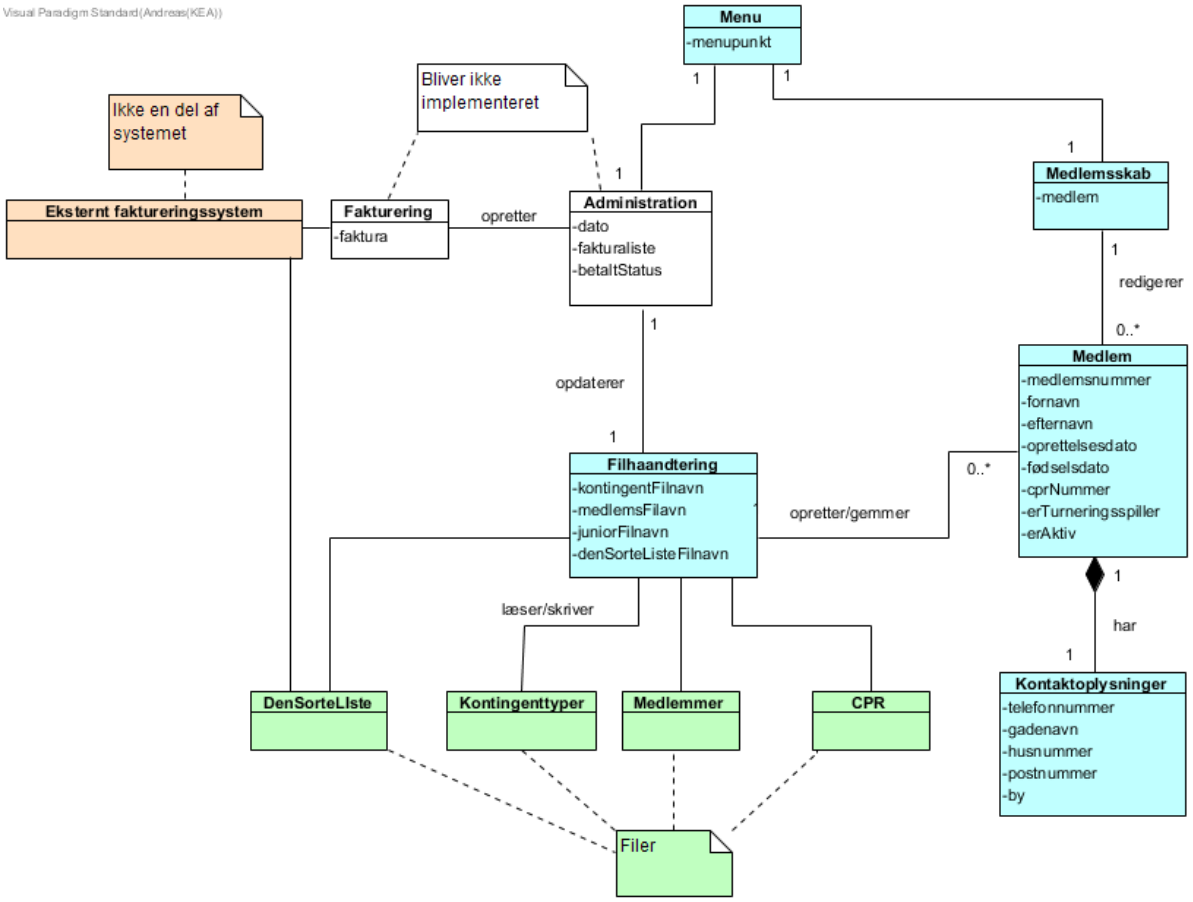
Det er dog ikke alle use cases og dermed heller ikke alle klasser, som blev implementeret i vores prototype (jf. afsnittet om afgrænsning af opgaven samt prototype og mulige udvidelser). De klasser, som ikke bliver implementeret, er markeret med hvid farve i domænemodellen.

Desuden har vi et eksternt faktureringssystem samt filer i databasen, som ikke er en del af systemet, selv om systemet har adgang til dem.

Opdelingen i klasser afspejler ansvarsfordeling mellem klasserne. Menu håndterer menuerne, som vises for brugeren, når man starter systemet. Medlemskab har ansvar for at håndtere medlemmer (bla. at oprette et medlem), som til gengæld har deres medlemsoplysninger i klassen Medlem. Desuden har vi klassen Filhåndtering, som har ansvar for at læse og skrive til filer.

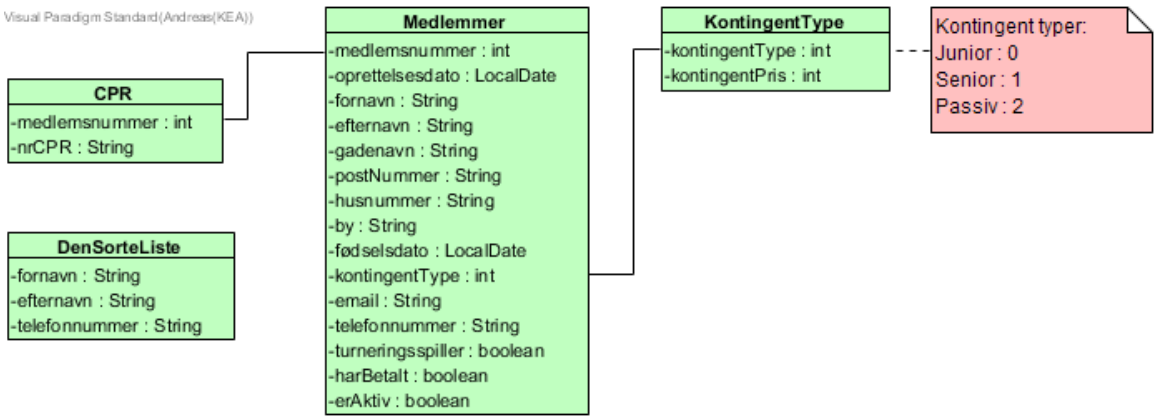
Forholdet mellem klassen Medlem og klassen Kontaktoplysninger indebærer composition, således at Kontaktoplysninger er en del af Medlem. Hvis et medlem bliver slettet, forsvinder også medlemmets kontaktoplysninger.

Domain Model (Hele gruppen)



Entity Relationship Diagram (Tekst af Andreas)

I domænemodellen er der beskrevet 4 filer, som systemet benytter til at håndtere data. Disse filer agerer som tabeller i en relationel database, hvor hver række er repræsenteret som en linje, og hver kolonne er separeret med tabulator (\t).



DenSorteListe holder oplysninger om alle medlemmer, der er bandlyst fra klubben pga. manglende betaling.

CPR holder medlemsnummer som foreign key og tilhørende CPR-nummer for klubbens juniormedlemmer.

Medlemmer har alle oplysninger om et specifikt medlem. Booleans bliver gemt som 0/1 i stedet for false/true. Medlemsnummer er primary key og holder kontingentType som foreign key.

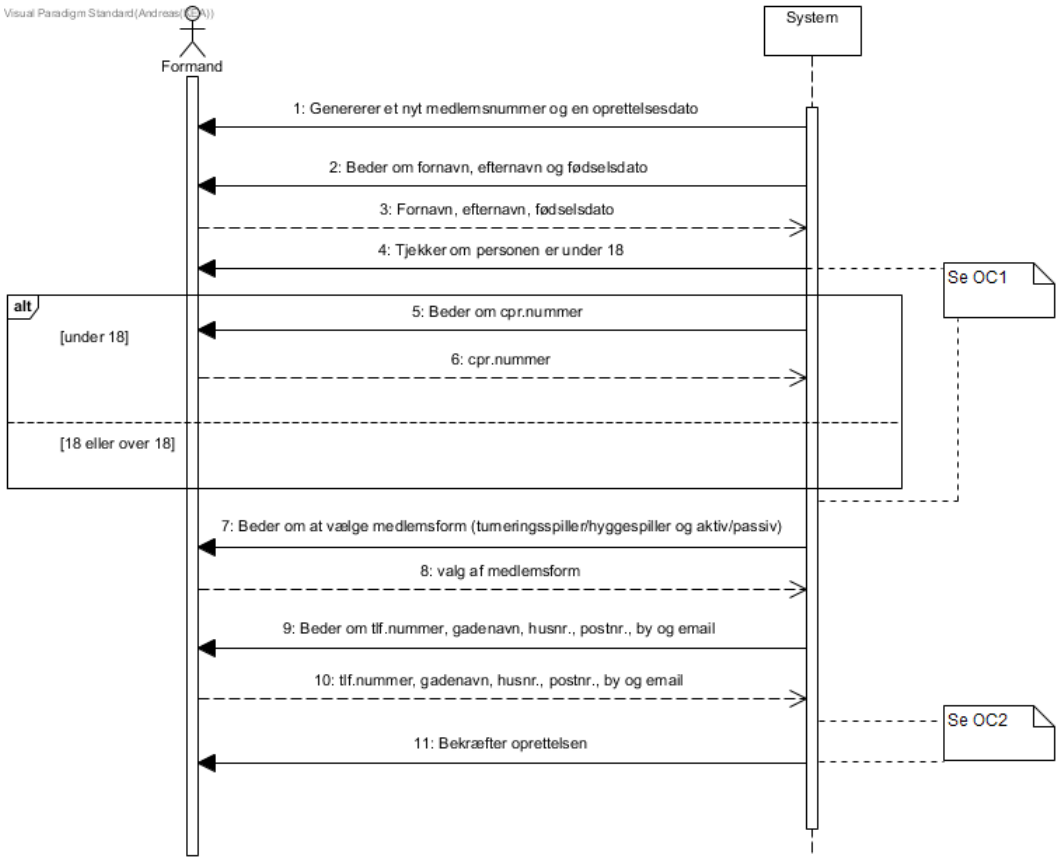
KontingentType gemmer de specificerede typer som 0,1 eller 2 og pris for tilhørende kontingent. Rabat for seniormedlemmer over 60 år bliver udregnet i systemet.

System Sequence Diagram (Tekst af Désirée)

System Sequence Diagram (SSD) er med til at vise et flow. Formålet er at illustrere de input og output, der er imellem brugere (aktør) og systemet for en bestemt begivenhed i et visuelt format.
Man læser det oppefra, hvor man øverst har placeret aktøren og systemet. Under dem finder man lange stiplede linjer, som bliver kaldt livliner. De linjer der strækker sig imellem dem, kaldes aktionslinjer, og de viser interaktionen mellem dem. Efter hver handling er blevet udført, bliver der vist svaret eller den næste handling.
Vores SSD tager udgangspunkt i vores use case 1 - oprette et nyt medlem.

System Sequence Diagram (Hele gruppen)

Use Case 1: Oprette et nyt medlem



Operation Contract (Tekst af Päivi)

OC1: 4. Tjekker om personen er under 18
Cross references: UC1: Oprette et nyt medlem
Preconditions: Aktøren har indtastet fornavn, efternavn og fødselsdato.
Postconditions:

- Systemet tjekkede, om personen var under 18 år. Hvis ja: systemet bad om et cpr-nummer.
- Aktøren indtastede et cpr-nummer, hvis personen var under 18.

OC2: 11. Bekræfter oprettelsen
Cross references: UC1: Oprette et nyt medlem
Preconditions: Systemet har genereret et nyt medlemsnummer og en oprettelsesdato. Aktøren har indtastet fornavn, efternavn, fødselsdato og cpr-nummer, hvis personen er under 18 år. Aktøren har valgt medlemsform og indtastet tlf-nummer, gadenavn, husnr., postnr., by og e-mail.
Postconditions:

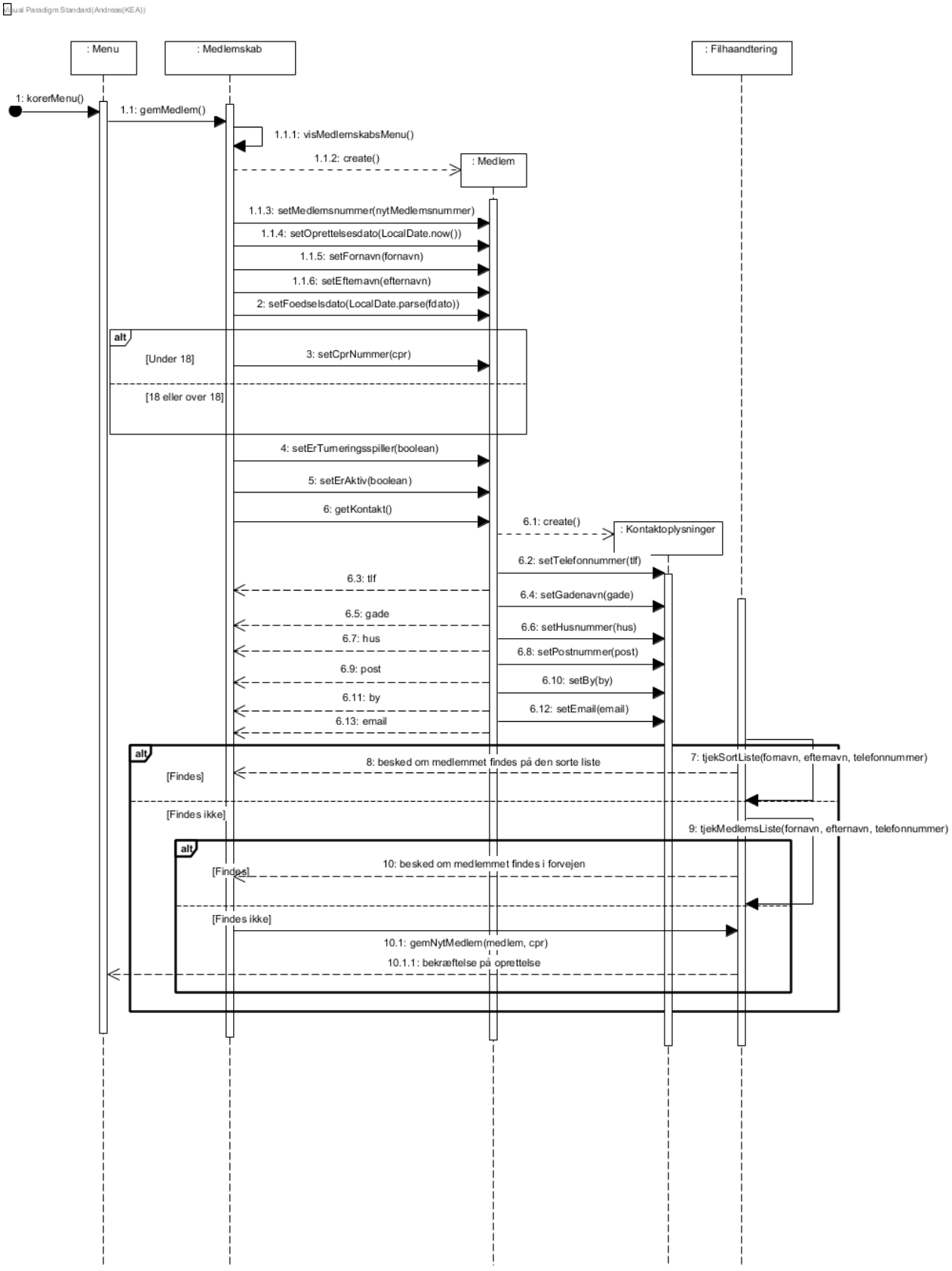
- Systemet tjekkede, om personen eksisterede i forvejen i systemet (extension til UC1). Hvis ja: aktøren fik besked om det, og oprettelsen blev afbrudt.
- Systemet tjekkede, om personen fandtes på den sorte liste (extension til UC1). Hvis ja: aktøren fik besked om det, og oprettelsen blev afbrudt.
- Hvis ingen af de to ovennævnte indtraf, blev medlemmet oprettet og aktøren fik en bekræftelse på oprettelsen.

Sequence Diagram (Tekst af Désirée)

Et Sequence diagram (SD) illustrerer en sekvens af handlinger, der sker i et system. Den viser de objekter og klasser, der er involveret, når en given use case bliver kørt med hvilken rækkefølge metodekaldene foretages i og hvilken retur-typer, de har.
De parallelle vertikale linjer (livliner), viser hvor længe objektet lever, og de vandrette pile, er meddelelser der bliver udvekslet mellem dem.
Man kan bruge SD til en slags sti for at illustrere, hvordan systemet skal respondere, og det gør det muligt at specificere et flow på en grafisk måde.
Vi har taget udgangspunkt til vores UC1 for at demonstrere, hvordan det virker i vores system.

Sequence Diagram (Hele gruppen)

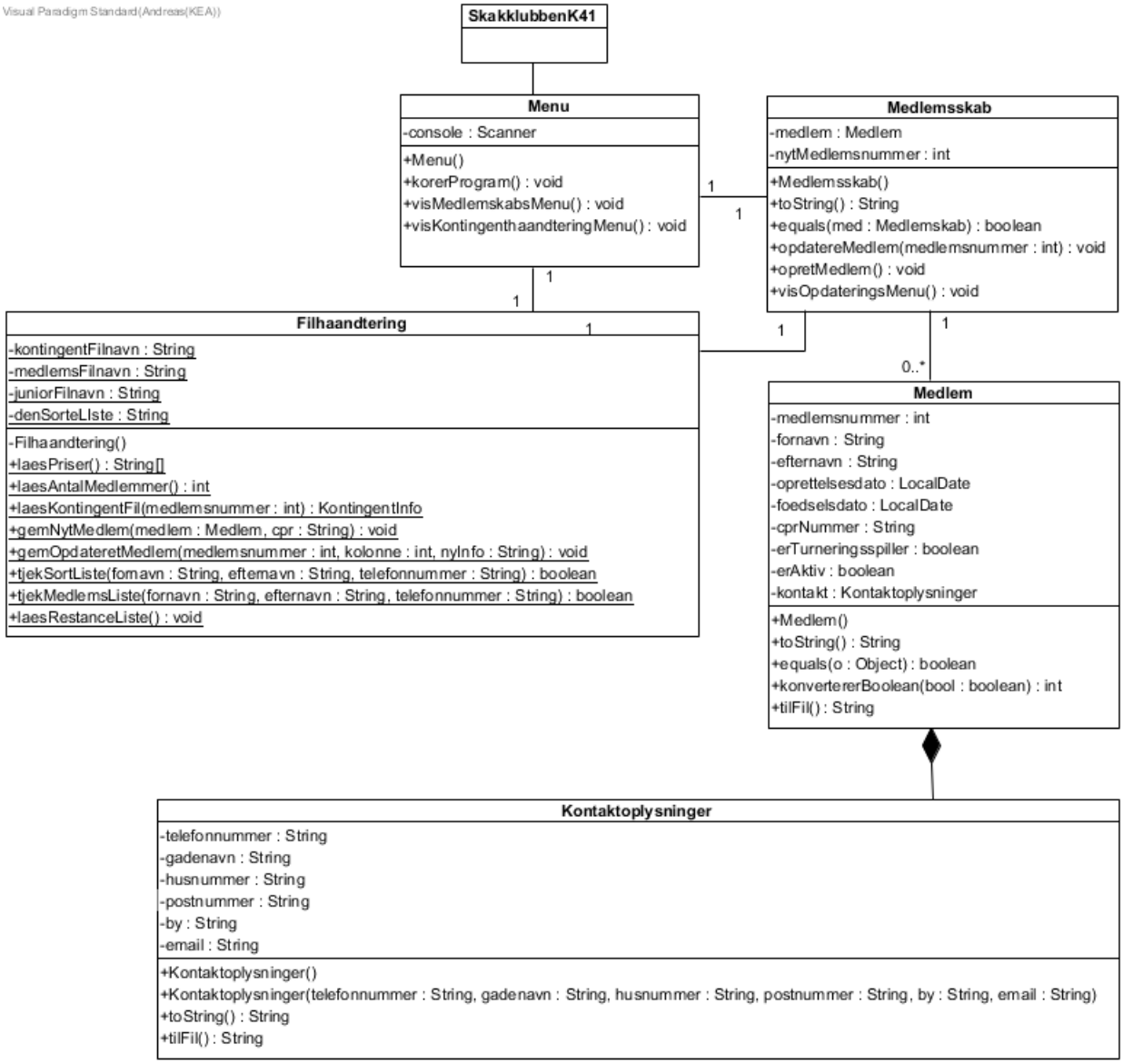
Use Case 1: Oprette et nyt medlem



Class Diagram (Tekst af Sofie)

Klassediagrammet er en visuel repræsentation af vores kode. Siden faktureringsdelen og administrationsdelen ikke implementeres i vores prototype, er de dele ikke vist i klassediagrammet, men kan dog findes i domænemodellen. Klassediagrammets attributter og metoder skal kun ses i forhold til de use cases, vi faktisk implementerer. Klassernes navne refererer til alle use cases.
Vi har valgt ikke eksplicit at vise getters og setters i klassediagrammet, da de er standarde i objektorienteret programmering med private attributter. Metoderne equals og toString er kun brugt i de klasser, hvor de er relevante i forhold til, hvor der håndteres oplysninger i vores system.

Class Diagram (Hele gruppen)



Prototype og mulige udvidelser (Tekst af Andreas)

Siden vores program skal ses om en prototype, har nogle af vores implementerede use cases visse begrænsninger i den egentlige kode. Disse begrænsninger er beskrevet her og kan altså ses som mulige udvidelser.

UC1: Man bliver bedt om at vælge, om man vil være hyggespiller eller turneringsspiller, selvom vores prototype ikke understøtter ratings. Dette sker, da vi gerne vil vise, at der er mulighed for at være turneringsspiller i klubben.
UC1 extension: vi har ikke har implementeret *a til fulde, og prototypen er derfor ikke sikret fuldkommen mod forkerte brugerindtastninger.
UC3: Det er lige nu ikke muligt at se, hvad det enkelte medlem skylder, da dette hører sammen med faktureringsdelen.
UC5: Opdatering af medlemsoplysninger. Denne use case er blevet afgrænset yderligere, hvor der i prototypen kun er mulighed for at ændre sit fornavn, efternavn eller telefonnummer.
UC6: Opdatering af medlemskab. Denne use case er også blevet afgrænset, så det kun er muligt at ændre sit medlemskab fra aktiv til passiv og omvendt. Vi har valgt ikke at gøre det muligt her i prototypen at kunne ændre sit medlemskab fra hyggespiller til turneringsspiller og omvendt, da vi ikke implementerer ratingdelen.

Konklusion (Tekst af Päivi)

I dette projekt har vi udviklet et administrativt system til Skakklubben K41 samt udarbejdet hertil knyttede analyser og diagrammer. Udgangspunktet for projektet var, at Skakklubben K41 ønskede at udvikle et administrativt system til at styre sine medlemsoplysninger, kontingenter og spillernes ratings for herigennem at kunne drive klubben på en mere professionel måde.

Missionen for klubben var på sigt at skabe fællesskab, skabe interesse og stimulere logisk tænkning hos såvel unge som voksne. Denne mission udmøntede sig i en mere konkret business vision, hvor klubben bl.a. har sat sig et mål om at vinde Danmarksmesterskabet i skak senest i 2023. Det ønskede administrationssystem er en løftestang for denne vision, da det vil hjælpe med at skabe overblik over kundetilgang og -afgang og dermed bistå til at udvikle klubbens virke på længere sigt.

Indledningsvis valgte vi i vores projektgruppe at afgrænse projektet. Afgrænsningen var baseret på gruppens vurdering af, hvor meget tid og hvor mange ressourcer (kompetencer og arbejdstimer), vi havde til rådighed til opgaven. Ud fra denne vurdering valgte gruppen at udvikle en prototype til et administrativt system med mulighed for udvidelse. I denne prototype valgte gruppen kun at implementere medlemsdelen, da vi anså denne for at være kernen i det administrative system.

En mere detaljeret afgrænsning findes i rapportens afsnit ”Afgrænsning af opgaven” og ”Krav”. Hertil kommer analysen af systemets funktionalitet og andre elementer i en såkaldt FURPS+ huskeliste, som findes i afsnittet ”FURPS+”.

Kravene til administrationssystemet bestod af de såkaldte CRUD-elementer: create, read,

update og delete. Kravene er specificeret i afsnittet "Krav". For at understøtte det administrative system skabte gruppen en database over medlemmer. Det administrative system betjenes gennem en menu.

I løbet af projektet foretog vi en yderligere afgrænsning af systemet – igen baseret på et helhedsskøn over, hvor megen tid og hvor mange ressourcer vi havde til rådighed. Denne afgrænsning resulterede i en beslutning om, at den færdige prototype skal leve op til følgende krav: oprette et nyt medlem (C), se hvilke medlemmer der er i restance (R) samt opdatere medlemmernes oplysninger (herunder bl.a. navn og medlemsform, U). De resterende krav ville man kunne levere til kunden efter at kunden har godkendt prototypen.

Gruppen udpegede ikke nogen projektleder, men styrede projektet i fællesskab ved hjælp af Trello-programmet som Kanban-værktøj. Denne arbejdsform gav alle lige muligheder for at deltage i projektstyringen, hvilket var en relevant del af læringsprocessen. Dette forløb er dokumenteret i bilagene med skærmbilleder af Trello, som vi har taget løbende.

Projektet blev udviklet som en UP-proces med hertil knyttede UML-analyseværktøjer. I den første fase (inception) blev kravene til projektet fastsat, use cases udpeget og den mest centrale use case beskrevet som fully dressed. Dette arbejde blev udført til dels på baggrund af mission og business vision, til dels ud fra tre forskellige analyser. Interessentanalysen gav et billede af aktørerne i projektet og SWOT-analysen definerede Skakklubbens styrker, svagheder, muligheder og trusler. Hertil kom en risikoanalyse, som udpegede mulige risikomomenter knyttet til projektet.

Dernæst gik gruppen over til den næste fase (elaboration), hvor hele analysedelen blev færdiggjort. I analysedelen blev resten af use cases kort beskrevet og der blev udarbejdet forskellige diagrammer til at visualisere use cases: Use Case Diagram, domænemodel, System Sequence Diagram samt to Operation Contracts. Normalt gennemgår man disse diagrammer sammen med kunden, sådan at kunden kan danne sig overblik over projektet og deltage i planlægningen, men da gruppen i virkeligheden ikke var i kontakt med kunden, blev dette ikke aktuelt.

I overgangen mellem den anden og den tredje fase (construction) blev der udarbejdet Sequence Diagram, som normalt ikke kommunikeres ud til kunden, idet det handler om koder. I denne UP-fase er vi ovre i designdelen, hvor gruppen udarbejdede et Entity Relationship Diagram, som viser oversigt over filerne i systemet, samt et klassediagram, som detaljeret viser en oversigt over koden i programmet. Til sidst kodede vi prototypen. Det skal dog nævnes, at fordi UP-processen er agil, blev de forskellige dele i opgaven løbende justeret, således at de afspejler den endelige prototype.

Projektet indeholder mange muligheder for udvidelse. For det første kunne man implementere alle use cases og alle funktioner, som er udeladt (jf. beskrivelsen af afgrænsningen). For det andet kunne gruppen bistå kunden med den sidste fase i UP-processen (transition), nemlig at overføre programmet til kundens system.

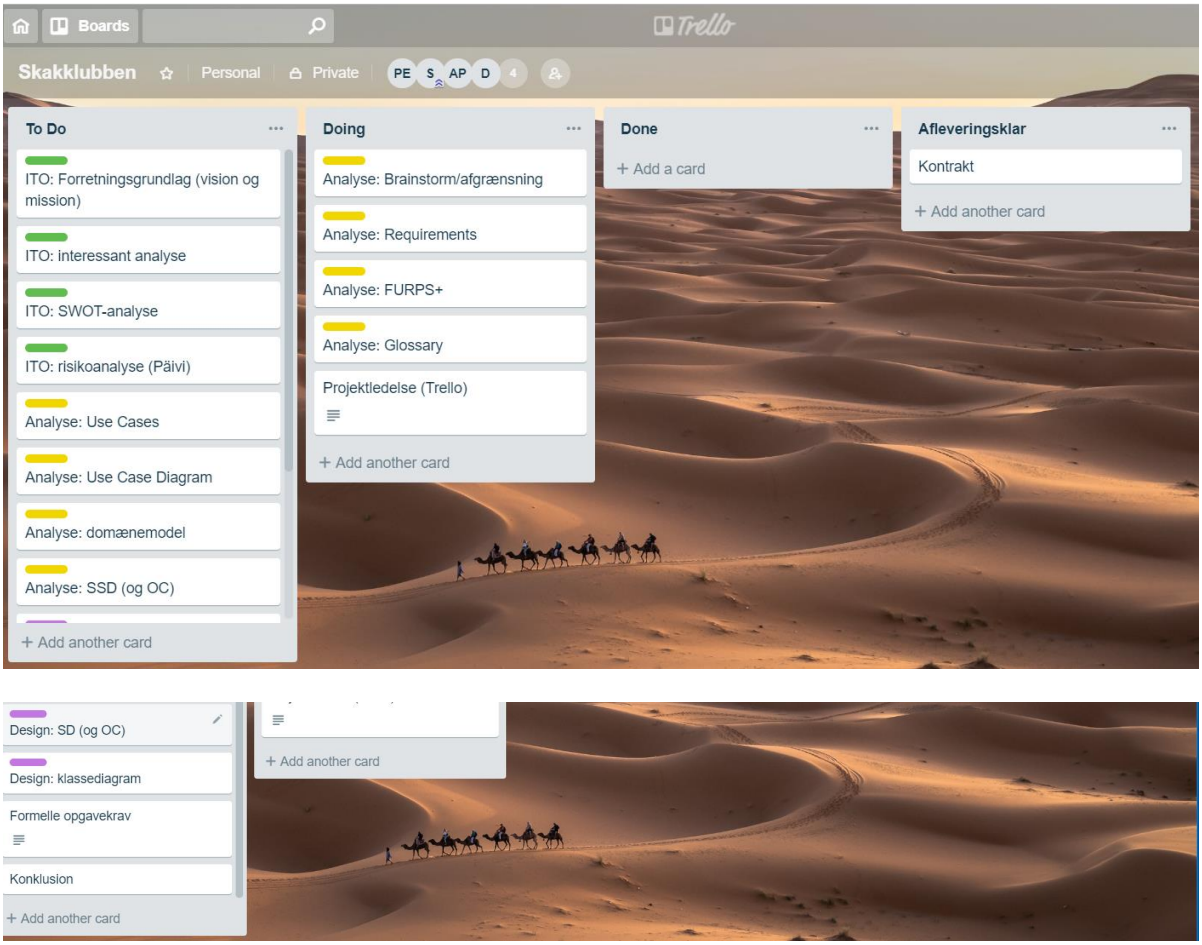
Bilag

Koden findes i *src* mappen og java doc findes i *docs* mappen inde i vores github repository.

Kanban board (Trello):

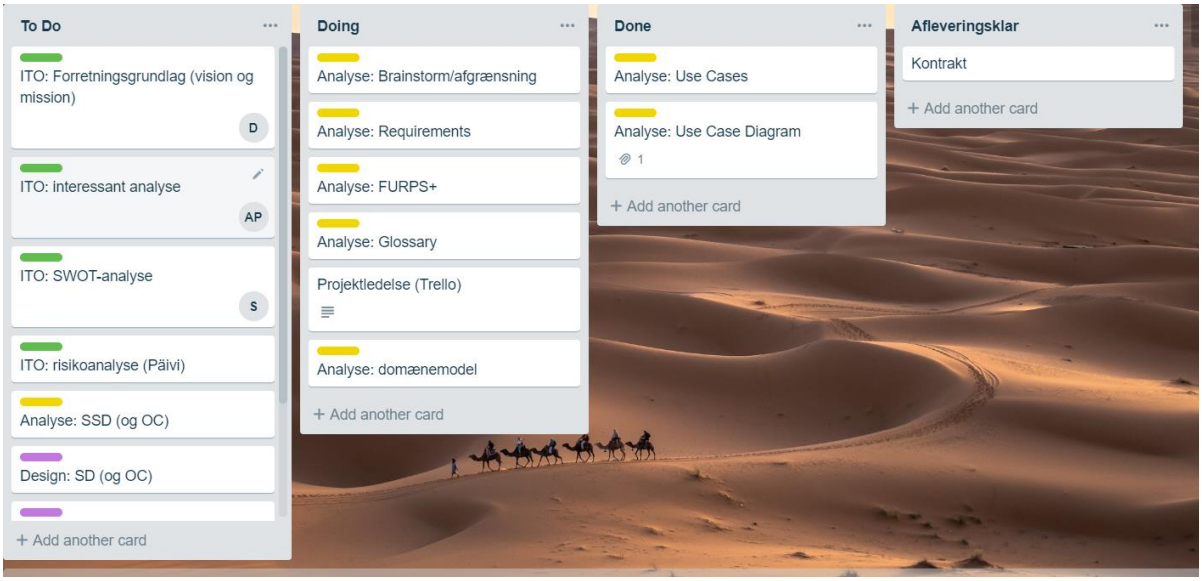
Bilag 1
Skærmbilleder af Trello

21.11.18

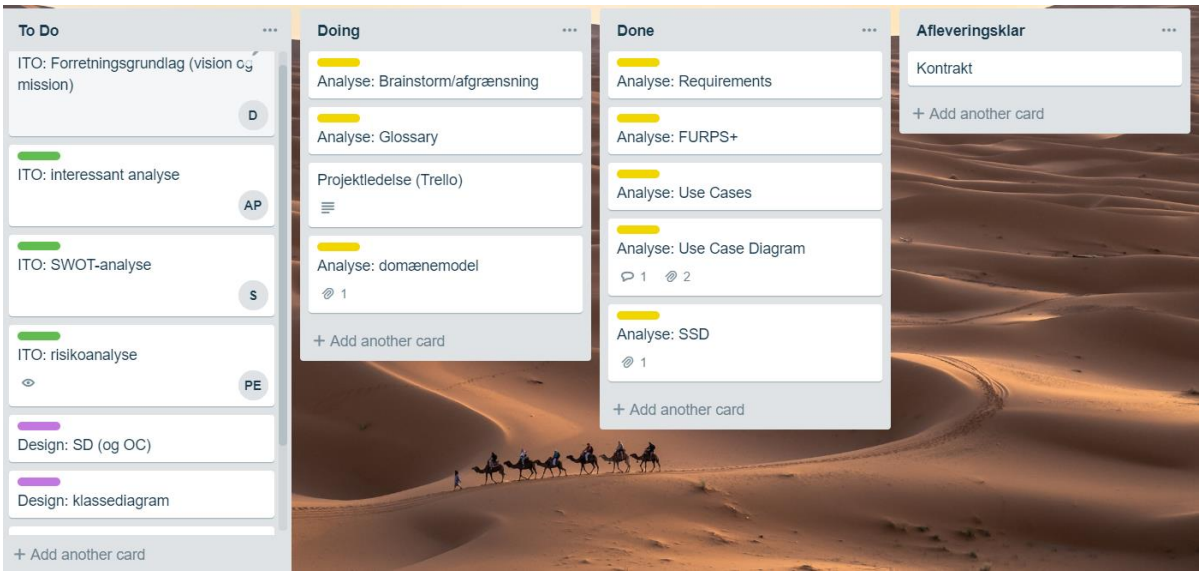


1

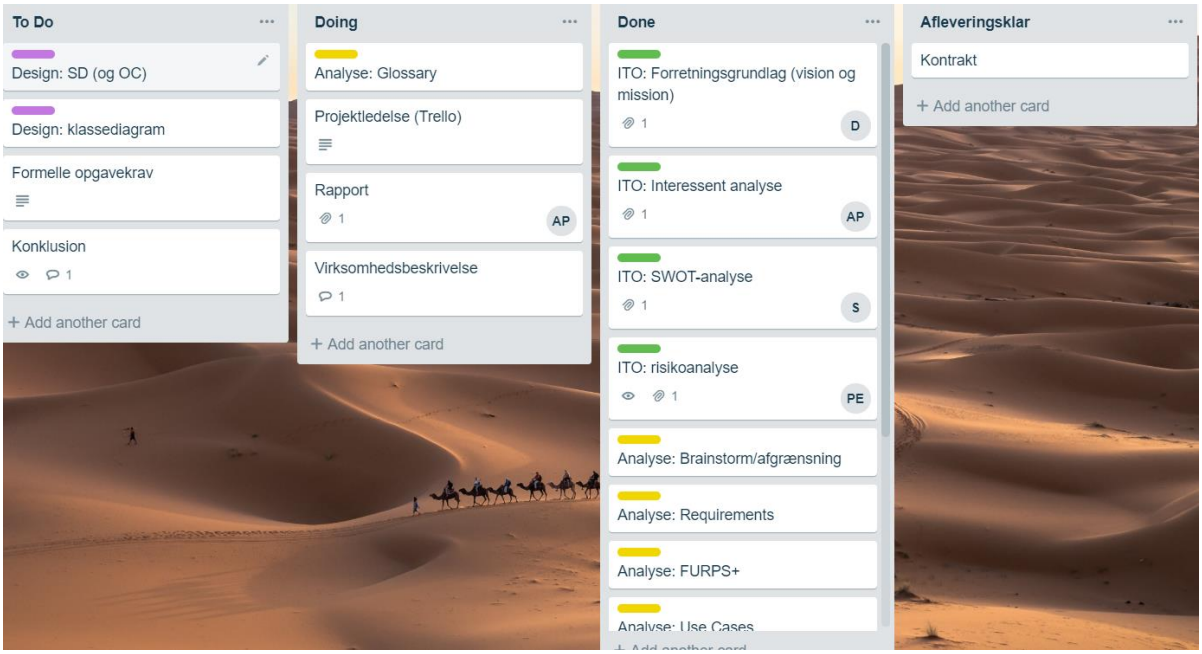
22.11.18


To Do
ITO: Forretningsgrundlag (vision og mission)
D
ITO: interessant analyse
AP
ITO: SWOT-analyse
S
ITO: risikoanalyse (Päivi)
Analyse: SSD (og OC)
Design: SD (og OC)
+ Add another card
Doing
Analyse: Brainstorm/afgrænsning
Analyse: Requirements
Analyse: FURPS+
Analyse: Glossary
Projektledelse (Trello)
Analyse: domænemodel
+ Add another card
Done
Analyse: Use Cases
Analyse: Use Case Diagram
1
+ Add another card
Afleveringsklar
Kontrakt
+ Add another card

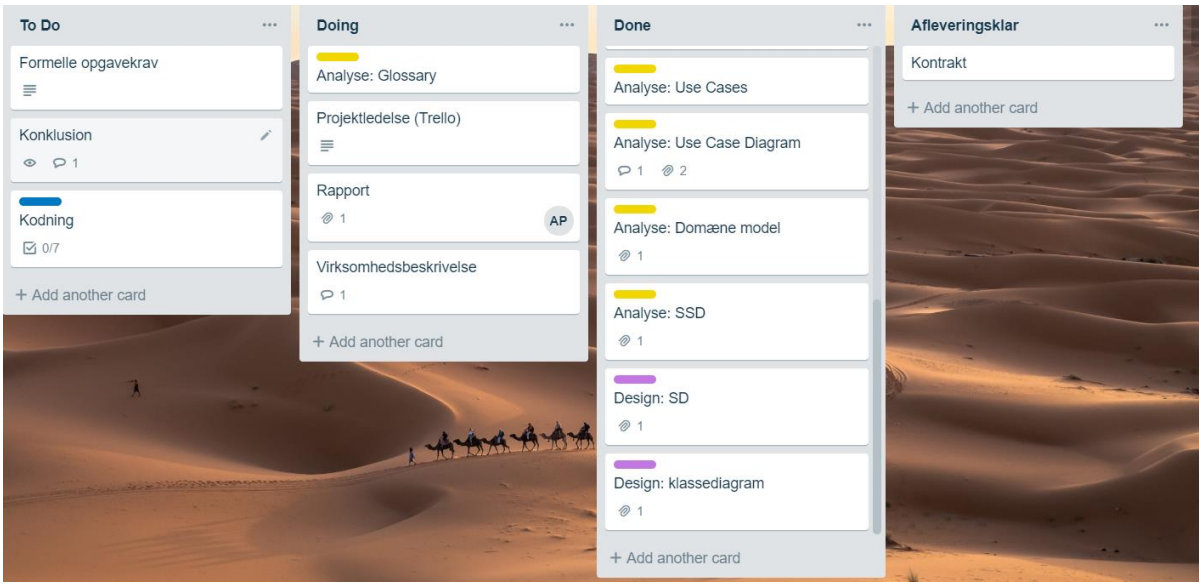
23.11


To Do
ITO: Forretningsgrundlag (vision og mission)
D
ITO: interessant analyse
AP
ITO: SWOT-analyse
S
ITO: risikoanalyse
PE
Design: SD (og OC)
Design: klassediagram
+ Add another card
Doing
Analyse: Brainstorm/afgrænsning
Analyse: Glossary
Projektledelse (Trello)
Analyse: domænemodel
1
+ Add another card
Done
Analyse: Requirements
Analyse: FURPS+
Analyse: Use Cases
Analyse: Use Case Diagram
1
2
Analyse: SSD
1
+ Add another card
Afleveringsklar
Kontrakt
+ Add another card

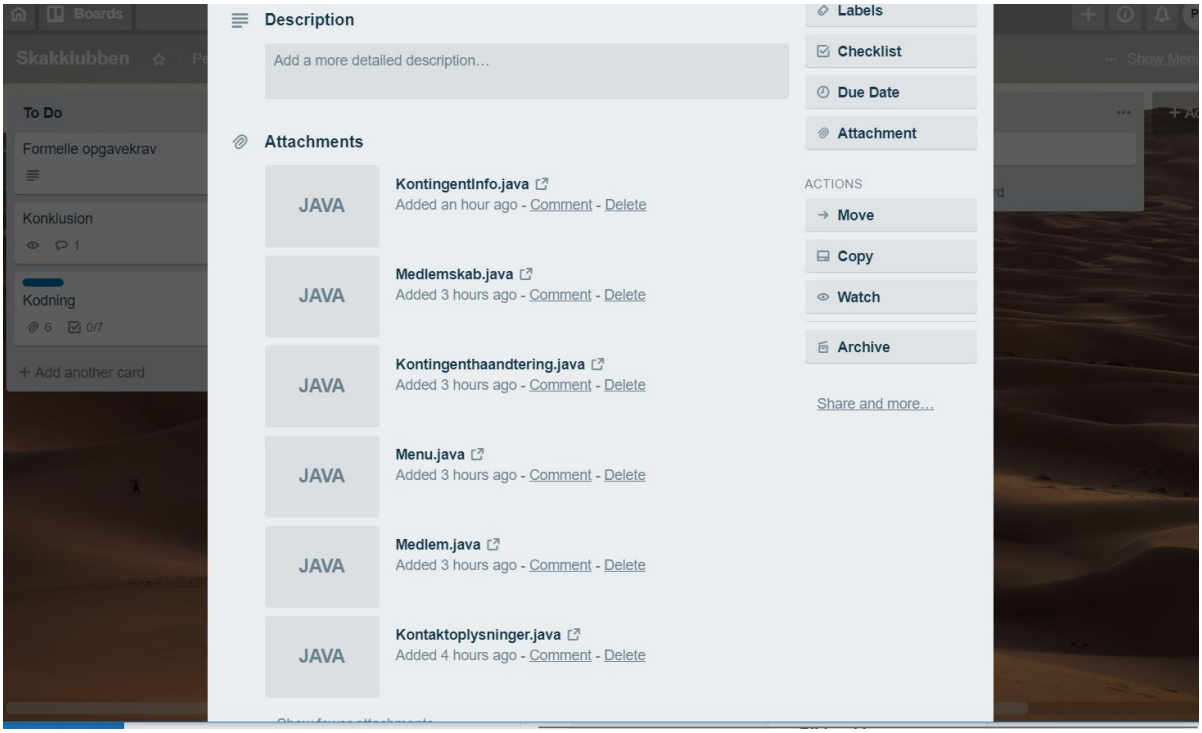
26.11.18



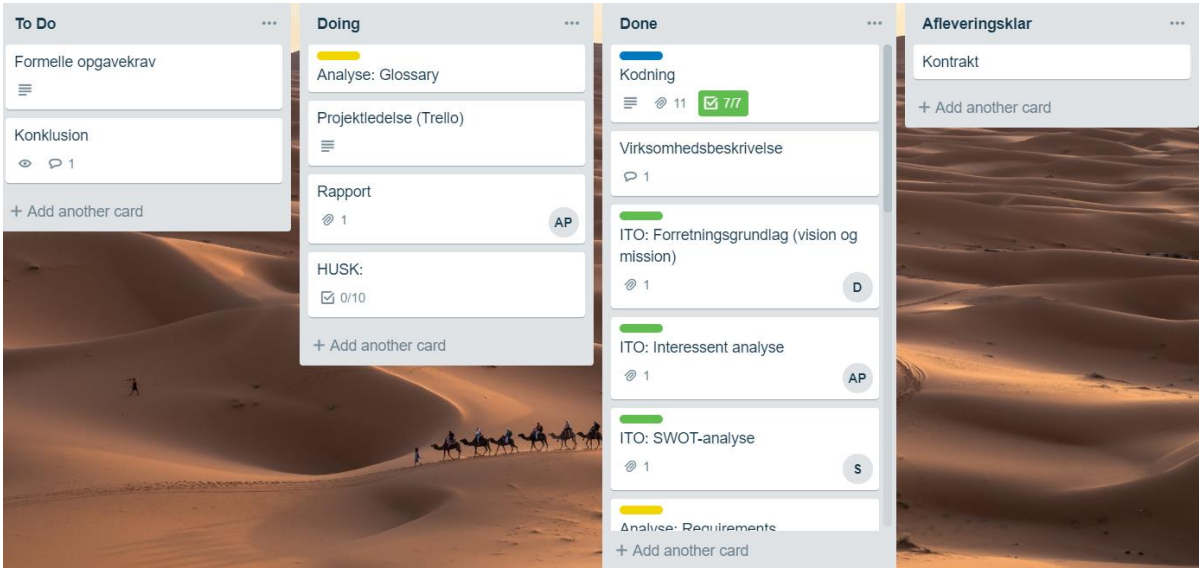
27.11.18



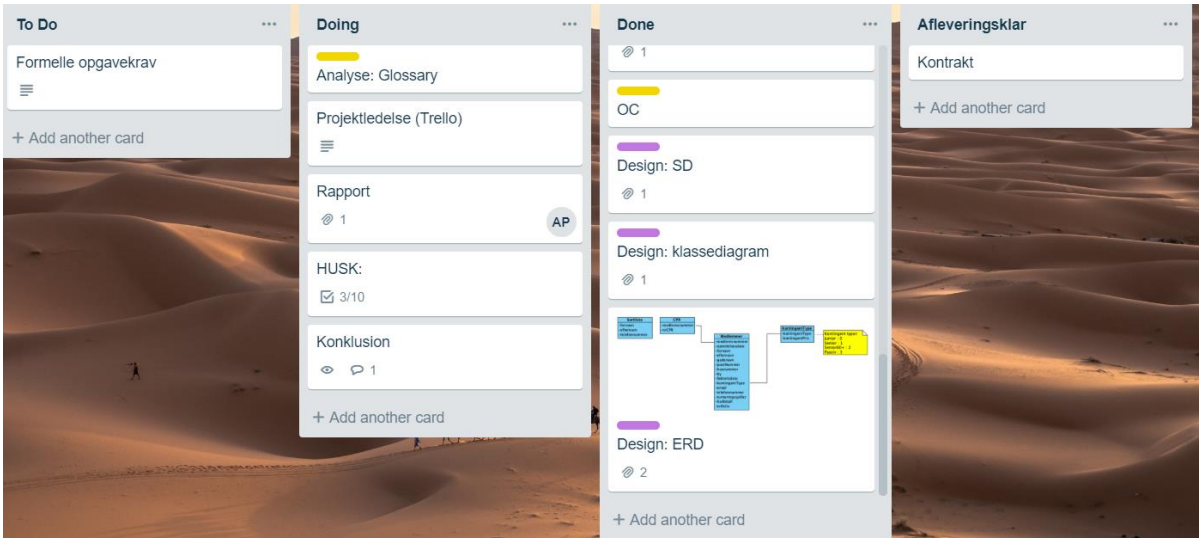
28.11.18 & 29.11.18



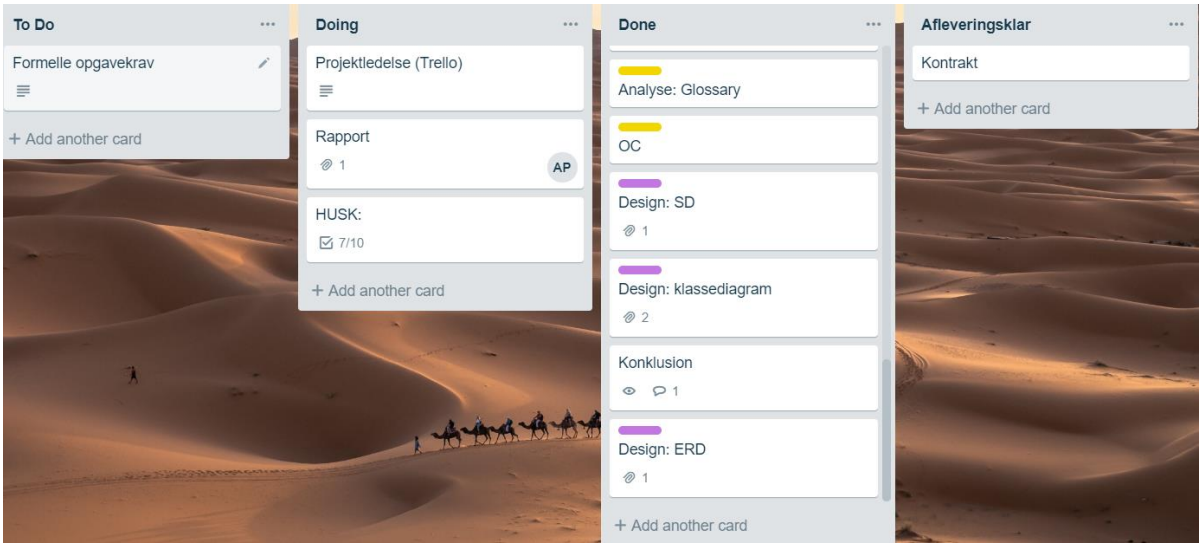
3.12.18



4.12.18

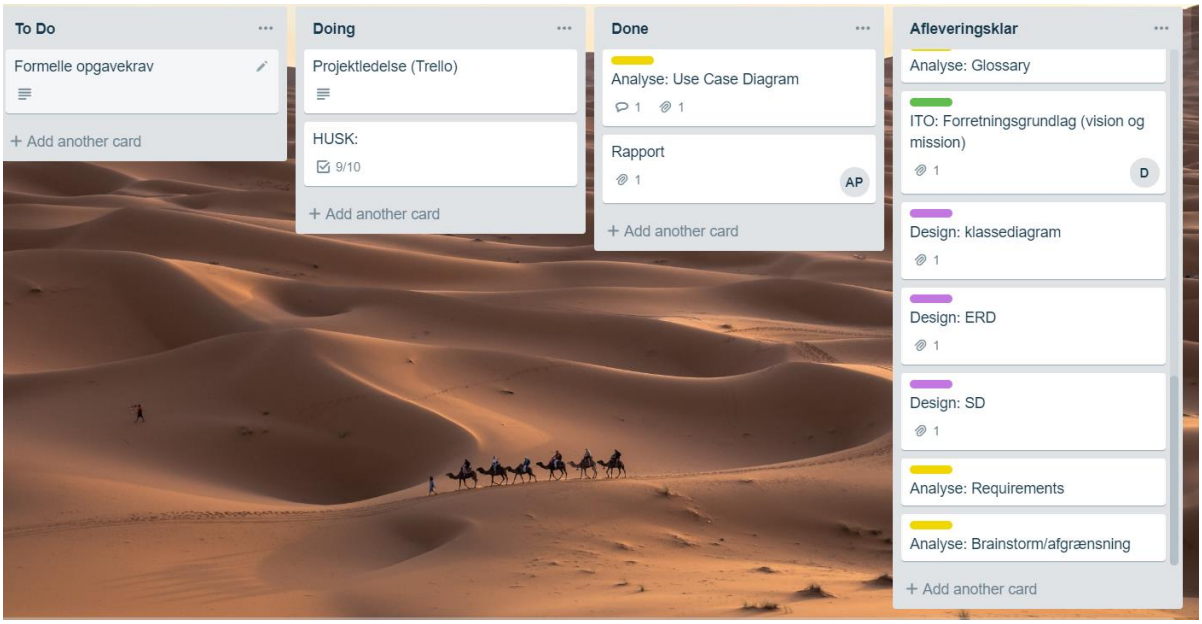


5.12.18



5

6.12.18


To Do
Formelle opgavekrav
+ Add another card
Doing
Projektledelse (Trello)
HUSK:
9/10
+ Add another card
Done
Analyse: Use Case Diagram
1
1
Rapport
1
AP
+ Add another card
Afleveringsklar
Analyse: Glossary
ITO: Forretningsgrundlag (vision og mission)
1
D
Design: klassediagram
1
Design: ERD
1
Design: SD
1
Analyse: Requirements
Analyse: Brainstorm/afgrænsning
+ Add another card

6