



University of
Southern Denmark

Udarbejdelse af et Credits Management System

UNIVERSITY OF SOUTHERN DENMARK

FACULTY OF ENGINEERING

MAERSK MC-KINNEY MOLLER INSTITUTE

2. SEMESTER PROJEKT SOFTWARE

KURSUS KODE - ST2-PRO

PROJEKT PERIODE: 01. FEBRUAR 2021 - 31. MAJ 2021

Gruppe ST01

Mathias Jeppesen Engmark

maeng20@student.sdu.dk

Oliver Heine

olhei20@student.sdu.dk

Anton Valdemar Dahlin Irvold

anirv20@student.sdu.dk

Nicklas Bruun Jensen

nickj20@student.sdu.dk

Christoffer Schurmann Krath

chkra19@student.sdu.dk

Kasper Stokholm

kasto16@student.sdu.dk

Vejleder

Henrik Lange

hela@mmmi.sdu.dk

II Titelblad

UNIVERSITY OF SOUTHERN DENMARK

FACULTY OF ENGINEERING

MAERSK MC-KINNEY MOLLER INSTITUTE

2. SEMESTER PROJEKT SOFTWARE

KURSUS KODE - ST2-PRO

PROJEKT PERIODE: 02. FEBRUAR 2021 - 31. JUNI 2021

Forfattere

Mathias J. Engmark _____

Oliver Heine _____

Anton V. D. Irvold _____

Nicklas B. Jensen _____

Christoffer S. Krath _____

Kasper Stokholm _____

III Resumé

Ved projektstart blev en case fra TV2 udleveret, som adresserede et direkte problem, virksomheden ønsker løst. Problemet omhandler deres nuværende måde at håndtere krediteringer, rulletekster, som optager 30 sekunder efter hvert udsendt program. TV2 ønsker på baggrund deraf et Credits Management System, som kan håndtere krediteringerne for TV2, og derved fjerne dem fra fjernsynet. Følgende problemstilling blev opstillet; *Hvor-dan kan man opstille et Credits Management System, som gør det muligt for TV2 at afskaffe krediteringstiden på programmer?* Problemformuleringen blev yderligere afgrænset til, kun at omhandle udviklingen af en prototype, inden for projektets faglige niveau, som blev adresseret ved Kapitel 2.3.

Der blev udarbejdet krav, for det system, der skulle udvikles, hvilke blev prioriteret gennem MoSCoW-metoden, og en analyse af projektets *must-have* krav blev efterfølgende udarbejdet, som resulterede i detaljerede brugsmønstre af kravene.

Implementeringen tog udgangspunkt i det udarbejdede design af systemnet og den dertilhørende database, som blev udarbejdet på baggrund af systemets krav og den dertilhørende analyse. For strukturering af projektudviklingen blev der gjort brug af Unified Process i en kombination med Scrum.

Der blev udarbejdet en række tests, som viste at projektets *must-have* krav virker tilstrækkeligt implementeret i forhold til den grundlæggende funktionalitet, men samtidig også, at der er plads til videreudvikling. Endvidere vurderes det, at resultatet af de udarbejdet test, kunne have set anderledes ud, hvis de var udført direkte på kunden, TV2, hvilket kan skabe muligheden for et mere brugerdefineret produkt. Ud fra de udarbejdede tests, vurderes det, at systemet kan betragtes som et muligt løsningsforslag, som kan bruges ved en eventuel videre udvikling af TV2's egentlige problem.

IV Forord

Denne rapport er udført som et 10 ECTS kursus for Mærsk Mc-Kinney Møller Institut i forbindelse med 2. semester Softwareteknologi. Grundet COVID-19 er det primære arbejde til projektet blevet udført online via Discord, Microsoft Teams og Zoom møder.

Rapporten er lavet som en dokumentation til et løsningsforslag ud fra en case udleveret af TV2, med hensigt på, at kunne opstille et Credits Management System, som kan afskaffe krediteringstiden på programmer. Projektet er udarbejdet med et mindre samarbejde med Morten Lehm fra TV2 gennem to kundemøder, som yderligere har klargjort casen. Rapporten forholder sig til det egentlige problem som kunden, TV2, har, og beskriver det arbejde, som gruppen har udført, for at opnå det endelige løsningsforslag.

Rapporten er udført, så den kan forstås af personer, som mindst besidder de fagligheder projektgruppen har opnået i løbet af andet semester på uddannelsen, men rapportens målgruppe er dog tiltænkt vejleder og censor, eftersom projektet er en del af et eksamsprojekt, og indirekte TV2, da rapportcasen bygger på deres problem.

Gruppen vil gerne takke Henrik Lange & Lone Borgersen for en yderst kompetent vejledning og undervisning, som har højnet det faglige aspekt for udviklingen af projektet. Endvidere skal der lyde et stort tak til Morten Lehm for et godt samarbejde i forbindelse med projektet. Endeligt skal der også lyde et tak til de testpersoner, som har hjulpet med udførelsen af test-afsnittet.

V Indholdsfortegnelse

1	Indledning	1
1.1	Introduktion	1
1.2	Baggrund	1
1.3	Problemformulering	2
1.4	Afgrænsning	2
2	Faglige Vidensgrundlag	3
2.1	Den udleverede case	3
2.2	Begrebsdefinition	3
2.3	Fagligt vidensgrundlag	4
3	Metode & Planlægning	6
3.1	Metode	6
3.2	Planlægning	6
3.3	Tidsplan	7
4	Krav	8
4.1	Overordnede krav	8
4.2	Detaljerede krav	11
5	Analyse	14
5.1	Statisk analysemodel	14
5.2	Dynamisk analysemodel - Adfærdsanalyse	21
5.2.1	Brugsmønsterrealisering af B11 Opdater kreditering på program .	21
5.2.2	Brugsmønsterrealisering af B13 Godkend krediteringer	23
5.2.3	Analyseklassediagram	26
6	Design	28
6.1	Design af Credits Management System	28
6.2	Design af database	29
7	Implementering	33
7.1	Implementering af B11 Opdater kreditering på program	33
7.2	Implementering af B13 Godkend krediteringer	36
7.3	SQL-Implementering	38
8	Test	41
8.1	Unittest	41

8.2	UP Brugsmønstertest	43
8.2.1	Test af brugsmønster B11 Opdater kreditering	43
8.2.2	Test af brugsmønster B13 Godkend program	46
8.3	Brugertest	49
9	Diskussion	51
10	Konklusion	53
11	Perspektivering	54
12	Procesevaluering	55
12.1	Samarbejdet i gruppen	55
12.2	Samarbejde med vejleder	55
12.3	Læringsprocessen	56
12.4	Projektarbejdsform	56
12.5	Hvad kunne være gjort bedre?	57
	Bibliografi	59
	Appendices	60
A	Typer af supplerende krav	61
B	Detaljeret brugsmønstre	61
C	Brugsmønsterrealiseringer	69
D	Fuld SQL-Script	76
E	Brugertest: Feedback	82
F	Brugervejledning	83
G	Logbog	101
H	Designklassediagram	101
I	Oversigt over kildekode	101
J	Udfyldt rapportKontrolskema	101
K	Inceptionsdokument	107

VI Læsevejledning

Denne rapport er opbygget kronologisk og kan derfor læses således. Det er ligeledes muligt at læse abstraktioner af rapporten, og stadig kunne opnå en generel forståelse for det givne emne. Det forventes på forhånd at læseren har et kendskab til programmeringssproget Java og dertilhørende JavaFX, endvidere en grundlæggende forståelse for SQL.

Kapitel 1 - Indledning, giver et indblik i projektets omfang, hvad der er valgt at arbejde med samt problemformulering og en afgrænsning af projektet.

Kapitel 2 - Faglig vidensgrundlag, indeholder en yderligere forklaring af projektets case, en begrebsafklaring for rapporten samt en beskrivelse af gruppens faglige niveau.

Kapitel 3 - Metode og planlægning, gives et indblik i den valgte projektmetode, samt hvordan planlægningen har forløbet.

Kapitel 4 - Krav, giver en forståelse af, hvordan kravspecifikationen er sammensat og prioriteret. Derudover findes der i kapitlet et brugsmønsterdiagram samt en detaljeret kravbeskrivelse. Afsnittet danner grundlag for indholdet i systemet.

Kapitel 5 Analyse, her gives en redegørelse af den statiske og dynamiske analyse, som efterfølgende anvendes konkret på rapportens forrige kapitel, Krav.

Kapitel 6 Design, giver læseren et indblik i gruppens designvalg og systemforståelse for udarbejdelsen af softwaren i form af et designklassediagram samt et diagram over data-basedesignet.

Kapitel 7 Implementering, omhandler implementeringen af udvalgt den grundlæggende funktionalitet ved to specificerede brugsmønstre samt implementeringen af databasen.

Kapitel 8 Test, omhandler gruppens softwaretest, bestående af Unittest brugsmønster-test og brugertest.

Kapitel 9 Diskussion, hvor resultaterne af testen og deres indvirkning for problemformuleringen diskuteres.

Kapitel 10 Konklusion, giver en sammenfattet konklusion på opgaven, hvor problemformuleringen bliver besvaret.

Kapitel 11 Perspektivering, giver indblik i, hvordan gruppen eventuelt kunne have grebet projektet anderledes an samt hvilke ændringer der kunne være lavet i projektet.

Kapitel 12 Procesevaluering, her gives en samlet gennemgang af gruppens arbejde gennem de to iterationer; fra projektstart til projektaflevering.

Kapitel 13 Bibliografi & Appendices, indeholder en samlet referenceliste og rapportens bilag.

VII Redaktionelt

Rapporten er skrevet som en dokumentation af projektet og dets resultater, hvor hvert afsnit er skrevet i et samspil mellem et til flere gruppemedlemmer. Hvert afsnit har fået tildelt et gruppemedlem(er) som *ansvarlig* samt *bidrag fra*, hvor den/de ansvarlig(e) har haft ansvaret for indholdet og udarbejdelsen af afsnittet, og fået yderligere hjælp af dem ved *bidrag fra*.

Afsnit	Ansvarlig	Bidrag fra	Kontrolleret af
Forside	Mathias	Anton	Alle medlemmer
Titelblad	Mathias	Kasper	Alle medlemmer
Resumé	Christoffer		Alle medlemmer
Forord	Anton	Oliver	Alle medlemmer
Indholdsfortegnelse	Alle medlemmer	Alle medlemmer	Alle medlemmer
Læsevejledning	Mathias		Alle medlemmer
Redaktionelt	Alle medlemmer	Alle medlemmer	Alle medlemmer
Indledning	Oliver	Mathias	Alle medlemmer
Faglig Vidensgrundlag	Christoffer		Alle medlemmer
Metode	Christoffer	Mathias	Alle medlemmer
Krav	Mathias, Oliver	Nicklas, Christoffer	Alle medlemmer
Analyse	Anton, Nicklas	Christoffer, Oliver	Alle medlemmer
Design	Kasper, Anton	Mathias, Christoffer	Alle medlemmer
Implementering	Oliver, Nicklas	Anton	Alle medlemmer
Test	Anton, Nicklas		Alle medlemmer
Diskussion	Oliver, Christoffer	Mathias	Alle medlemmer
Konklusion	Anton, Oliver		Alle medlemmer
Perspektivering	Christoffer, Kasper	Anton, Oliver	Alle medlemmer
Procesevaluering	Kasper	Christoffer, Mathias	Alle medlemmer
Bibliography	Alle medlemmer	Alle medlemmer	Alle medlemmer
Appendices	Mathias	Anton, Oliver, Nicklas	Alle medlemmer

Tabel 1 Ansvarsområde for hvert individuelt medlem samt bidrag fra andre gruppemedlemmer

KAPITEL 1

Indledning

Efter hver udseendelse hos TV-selskabet TV2, allokeres der 30 sekunder til rulletekster. Det resulterer i en nødvendig prioritering af, hvilke krediteringer, som skal vises i løbet af de 30 sekunder, hvilket TV2 gerne vil imødekomme. Forslaget lyder på et Credits Management System, som skal samle alle krediteringerne ét sted, og sikre en mere effektiv arbejdsproces samt at fjerne omtalte prioritering.

1.1 Introduktion

Projektet omhandler udviklingen af et Credits Management System for TV2. Systemet skal være med til at håndtere TV2's egne krediteringer, rulleteksterne, og skal på den måde sikre mere skærmtid til anden relevant materiale samt mindre manuelt arbejde fra TV2. Systemet udvikles som en prototype af det endelige produkt, og vil arbejde sammen med en simpel database, som skal sikre persisteringen af dataen i systemet.

1.2 Baggrund

Når et program er blevet udsendt, skal der vises rulletekster for det given program. Hos TV2 har de en maksimum-længde på rulleteksterne på 30 sekunder, hvilket resulterer i, at krediteringerne ofte skal prioriteres. Den nuværende måde at indskrive krediteringer foregår gennem indsending af et dokument, udarbejdet af en producer fra et given program, til en af TV2's egne folk, som indskriver krediteringerne til deres system, hvorefter det vises på fjernsynet som rulletekster. Omtalte arbejdsmåde resulterer i spild af ressourcer gennem dobbeltarbejde, samtidig med, den omtalte prioritering af krediteringerne. På baggrund deraf ønsker TV2 at få udarbejdet et Credits Management System, som skal erstatte de nuværende rulletekster ved TV2's egne programmer, og samle dem ét sted. Igennem TV2's egne beregninger, estimerer de, at de kan forøge deres årlige omsætning

med helt op til 60 millioner danske kroner, og samtidig sikre at alle krediteringer kan prioriteres lige højt, og at det automatiserer arbejdsbyrden for TV2. Ydermere frigøres der 30 sekunder efter en del af deres programmer, som kan blive brugt til anden relevant materiale.

1.3 Problemformulering

Ved inceptionsdokumentet blev der udarbejdet en problemformulering på baggrund af den udleveret case, se Bilag K Kapitel 1.4. Problemformuleringen lyder således: *Hvordan kan man opstille et Credits Management System, som gør det muligt for TV2 at afskaffe krediteringstiden på programmer?*

Hvortil der blev udarbejdet følgende underspørgsmål:

- Hvordan kan man lave et softwaresystem, som håndterer forskellige brugers roller og muligheder i et Credits Management System?
- Hvordan kan en database til et Credits Management System opsættes?
- Hvordan kan man lave en brugergrænseflade til et Credits Management System?
- Hvordan kan man lave et notifikationssystem, der underretter de relevante brugere om ændringer i Credits Management Systemet?

1.4 Afgrænsning

Projektet vil blive udarbejdet i forbindelse med følgende afgrænsning:

Credits Management Systemet vil blive udarbejdet som en prototype i overensstemmelse med de faglige kundskaber erhvervet igennem 2. semester af softwareteknologi uddannelsen, redegjort i Kapitel 2.3, og vil altså ikke være et fuldendt produkt. For persistering af data, vil der blive udarbejdet en PostgreSQL database, med tilhørende SQL-script, som vil ligge lokalt sammen med programmet.

KAPITEL 2

Faglige Vidensgrundlag

Forståelse af den udleverede case, begrebsdefinition samt gruppens faglige grundlag for projektet vil blive adresseret.

2.1 Den udleverede case

Casen, som blev udleveret af TV2, beskriver udviklingen af et krediteringssystem der har til formål at flytte arbejdsbyrden vedrørende indskrivning af krediteringer fra TV2, og ud til de individuelle producere. Udover dette er meningen også, at systemet skal opbevare krediteringer således, at TV2 ikke behøver vise disse i slutningen af hvert program. Dette resulterer i en frigørelse af 30 sekunder til at vise andet relevant indhold.

2.2 Begrebsdefinition

- Credit (Kreditering)
 - En credit er en kreditering for noget arbejde i forbindelse med et program.
Eksempelvis: Jens Jensen - Lydmand.
- Program
 - Et program er, i projektets sammenhæng, noget som TV2 sender. Dette kan både være en episode fra en TV-serie eller en transmission.
- Transmission
 - En transmission er en udsendelse der ikke består af *kronologiske*¹ afsnit. Eksempelvis nyhederne.

¹Der tages forbehold for at selve serien ikke er kronologisk

- TV-series
 - En TV serie er en serie af afsnit i en kronologisk rækkefølge. Eksempelvis badehotellet.
- Guest (Gæst)
 - En Gæst i projektets sammenhæng, er en som ikke er logget ind, og derved har færre muligheder end eksempelvis en Producer eller Systemadministrator.
- User (Bruger)
 - En User i projektets sammenhæng, kan betragtes som en almindelig bruger som kan logge ind. Ikke at forveksles med User-klassen i Java-koden.
- Producer
 - En producer er en producent oprettet i systemet af en systemadministrator, og er dem, som primært opretter programmer og krediteringer i systemet. Yderligere kan de opdatere krediteringer og programmer de selv har oprettet.
- Happy day
 - At arbejde ud fra happy day menes der, at der kun ses på det givne emne, hvor alt går godt, og der abstraheres fra eventuelle fejl eller alternative hændelser.
- Teamprogrammering
 - Samme som det velkendte parprogrammering, bare som en hel gruppe.

2.3 Fagligt vidensgrundlag

I løbet af projektets udarbejdelse, vil gruppen erhverve sig erfaring fra 2. semester af ingeniøruddannelsen softwareteknologi, som kan kategoriseres indenfor følgende:

- Computersystemer
- Problemorienteret projektarbejde
- Statistisk dataanalyse
- Objekt Orienteret Programmering
- Videregående Objektorienteret Programmering (VOP)

- Grundlæggende Software Engineering (GSE)
- Data Management (DM)

Af overstående har følgende været de primære færdigheder i dette projekt: *DM*, *GSE* samt *VOP*. Viden tilsvarende den gruppen har erhvervet eller bedre, er nødvendigt for at opnå forståelse af projektet. Projektet er derudover struktureret omkring Unified Process & SCRUM, som blev addresseret ved inceptionsdokumentet, se Bilag K Kapitel 2 & 6.1. Det er samtidig vigtigt at kende til disse processer for at få fuld forståelse for projektet.

KAPITEL 3

Metode & Planlægning

Projektgruppen har anvendt en kombination af Unified Process(UP) og Scrum for strukturering af projektet, hvor Scrum blev en del af projektet ved elaborationsfasen i UP, mens en tidsplan også blev defineret ved et Gantt-diagram. Ydermere er der anvendt par- & teamprogrammering for implementeringen af systemet.

3.1 Metode

I løbet af projektet er der anvendt en kombination af UP og Scrum, hvilken er beskrevet i inceptionsdokumentet, se Bilag K Kapitel 2 & 6.1. Til implementering er der anvendt par- & teamprogrammering. Undervejs har gruppen sørget for at skiftes til at programmere, og fortælle hvad som skulle skrives. Dette har sikret at alle har skrevet kode undervejs.

3.2 Planlægning

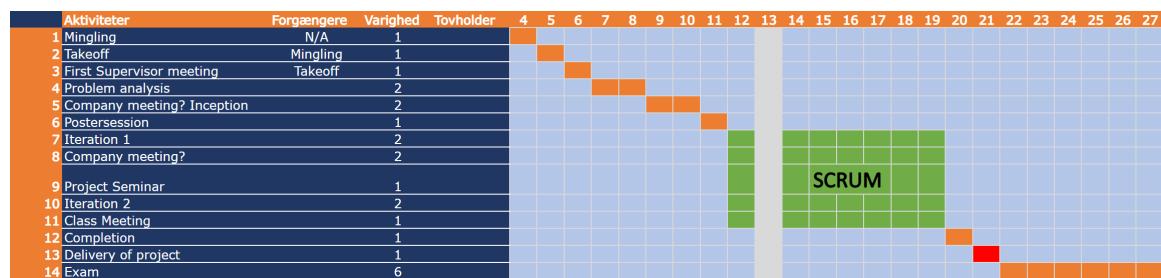
Som nævnt i Afsnit 3.1, er der blandt andet benyttet Scrum i løbet af projektet. Eftersom gruppen ikke har haft den største erfaring med brugen af Scrum, er det også grunden til, det ikke nødvendigvis er brugt helt optimalt og helt efter bogen. Gennem projektarbejdet har gruppen gjort brug af et Kanban board via platformen Jira, hvilket har været med til at give gruppen et bedre overblik over de arbejdsopgaver, som skulle laves i det pågældende sprint/iteration. Derudover har gruppen gjort brug af Daily Scrum, hvor det har været relevant.

Gruppen har valgt MoSCoW metoden til prioritering af, hvilke opgaver der har været de mest essentielle for projektet. Den prioritering blev også overført til gruppens backlog, - som ligger på Jira - hvor opgaverne blev valgt ud fra. Projektet i sin helhed har været opdelt i to iterationer, men de har ikke været yderst skelsættende for projektets forløb. Igennem 1. iteration blev størstedelen af softwarens logik, i henhold til prioriteringen,

implementeret, og der blev gjort brug af en midlertidig database i form af .txt-filer. Ved 2. iteration blev persistenslaget udskiftet fra at dække over .txt-filer, til at dække over en reel database, som blev skrevet ved brug af PostgreSQL. Årsagen til iterationerne er hovedsageligt grundet gruppens undervisning ved DM og GSE adresset ved Kapitel 2.3.

3.3 Tidsplan

I inceptionsfasen blev der udarbejdet et Gantt diagram, som havde til hensigt at danne en foreløbig tidsplan for projektet, som gruppen kunne forholde sig til. Den grønne sektion i diagrammet, Figur 3.1, viser perioden, hvor Scrum blev benyttet frem for Gantt diagrammet, da Scrum ikke fungerer sammen med planlægningen i Gantt, som til dels bliver belyst ved Bilag K Kapitel 6.1. Den hovedsagelig årsag ligger dog til grund i henhold til måden hvorpå Gantt og Scrum fungerer. Gantt fokuserer på slutmålet, hvor Scrum fokuserer på forbedringer fra det forrige sprint[1].



Figur 3.1 Gantt-diagram lavet for projektet under inceptionsfasen.

KAPITEL 4

Krav

Projektet har fulgt procesmodellen Unified Process, hvor der som et led i inceptionsfasen blev udarbejdet en rapport, inceptionsdokumentet Bilag K, som blandt andet indeholder projektets overordnet krav og prioritering af dem samt et brugsmønsterdiagram over, hvilke aktører som gør brug af hvilke brugsmønstre. Elaborationsfasen har efterfølgende taget udgangspunkt i resultaterne fra inceptionsfasen, hvor blandt andet implementeringen tog udgangspunkt i den prioritering som blev udarbejdet ved inceptionsfasen.

4.1 Overordnede krav

Den tidligere udarbejdet rapport fra inceptionsfasen, inceptionsdokumentet, beskriver blandt andet projektets overordnet krav, se Bilag K Kapitel 3, hvor prioriteringen, som bygger på MoSCoW metoden, af omtalte krav, er medtaget i Tabel 4.1, dog er manglende krav tilføjet og der tages forbehold for omprioritering og omnummerering sammenlignet med inceptionsdokumentet, se Bilag K Kapitel 5. Opdateringen af krav og brugsmønstre skyldes en ny erhvervet viden efter påbegyndt implementering. Tabellen repræsenterer hvordan projektets implementering er forløbet i forhold til prioriteringen, og viser hvilke brugsmønstre, som kan forventes implementeret.

ID	Navn	Prioritet
B10	Oprette et program med kreditering	M (1.)
B11	Opdatere kreditering	M (2.)
B13	Godkend program	M (3.)
B17	Opdatere program	M (4.)
B04	Logge ind	M (5.)
S03	Adgangsrettigheder	M (6.)
B06	Logge ud	M (7.)
B16	Slette kreditering	M (8.)
B18	Slette program	M (9.)

B01	Søg og gennemse kreditering	M (10.)
B05	Oprette en bruger	S (11.)
S02	Opdatering af krediteringer	S (12.)
B12	Eksportere kreditering	S (14.)
S01	Dataeksport	S (15.)
B14	Ændre bruger	S (16.)
S04	Brugervenlig-Interface	S (17.)
B15	Ændre adgangsrettigheder for anden bruger	C (18.)
B07	Gemme en kreditering som favorit	C (19.)
B08	Tilgå favorit kreditering	C (20.)
B02	Se egne kontooplysninger	C (21.)
B03	Redigere egne kontooplysninger	C (22.)
B09	Abonnere på en krediteret person	W (23.)

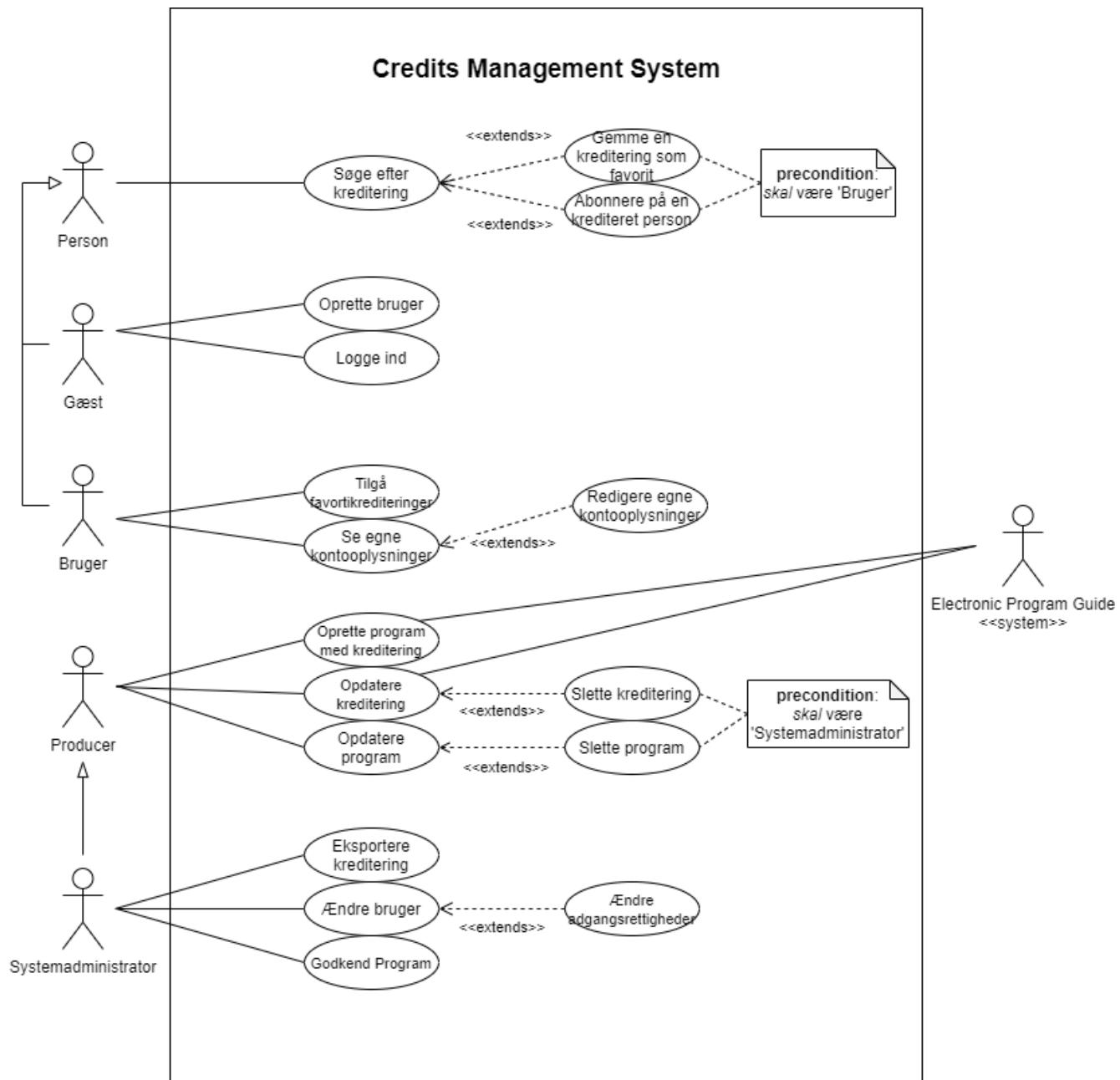
Tabel 4.1 MoSCoW-prioritering over projektets krav.

Projektet har opstillet enkelte supplerende krav, hvilke definerer implementeringsvilkår, som ved implementeringen vil følge prioriteringen ved Tabel 4.1. De supplerende krav er opstillet i Tabel 4.2, og tager udgangspunkt i Figur 1 Bilag A. De supplerende krav, blev også adresseret ved inceptionsdokumentet, se Bilag K Kapitel 3 afsnit 3.

Supplerende krav			
ID	Type	Navn	Beskrivelse
S01	Data integrity	Dataeksport	Krediteringen, som eksporteres, skal være utsynlig og kvalitetssikret.
S02	Manageability	Opdatering af krediteringer	Når en kreditering opdateres, skal denne kreditering stadigvæk vises og systemet skal virke.
S03	Security	Adgangsrettigheder	Kun aktører med rollen "Producer" eller "Systemadministrator" har adgang til information om krediteret personer, som de krediterede personer ikke har lyst til at oplyse til andre.
S04	Usability	Brugervenlig-Interface	Credit management systemets interface, skal være brugervenligt og intuitivt at forstå/bruge.

Tabel 4.2 Supplerende krav

Ydermere blev der, ved inceptionsdokumentet, opstillet et brugsmønsterdiagram, hvilket tager udgangspunkt i de identificerede aktører og deres brugsmønstre, se Bilag K Kapitel 3. Brugsmønsterdiagrammet, Figur 4.1, er udarbejdet som et detaljeret brugsmønsterdiagram, for at skabe en bedre forståelse af systemets opfattelse.



Figur 4.1 Brugsmønsterdiagram over Credits Management System

4.2 Detaljerede krav

Brugsmønster B11 og B13, som dækker over *opdater kreditering & godkend program*, vil blive yderligere detaljeret i Tabel 4.3 & 4.4. De to brugsmønstre vil også være dem, som gruppen vil fokusere på i resten af rapporten, og være dem, som der bliver brugt som eksempler. Disse brugsmønstre er valgt, da de var højt prioriteret i MoSCoW, se 4.1. Brugsmønstrene er udarbejdet på baggrund af inceptionsfasens overordnet brugsmønster, se Bilag K Kapitel 3, og Tabel 4.3 & 4.4 kan betragtes som de detaljerede brugsmønstre. De resterende detaljerede brugsmønster udarbejdet i elaborationsfasen kan ses i Bilag B.

Brugsmønster: Opdater kreditering	
ID: B11	
Primære aktører: Producer & Systemadministrator. (Omtales aktør fremover)	
Sekundære aktører: Ingen	
Kort beskrivelse: En producer skal kunne opdatere krediteringsoplysningerne for et program de ejer.	
Prækonditioner (Pre-conditions):	<ul style="list-style-type: none"> • Programmet skal have en oprettet kreditering i systemet. • Aktøren har valgt et program.
Hovedhændelsesforløb (main flow):	<ol style="list-style-type: none"> 1. Brugsmønstret starter når en aktør ønsker at opdatere kreditering for et program i systemet 2. Aktøren vælger en kreditering fra programmet 3. Aktøren trykker på knappen opdater kreditering 4. Systemet viser den valgte kreditering for programmet 5. Aktør opdaterer felter med nye krediteringsoplysninger 6. Aktør trykker “opdater kreditering” 7. Systemet opdaterer den valgte kreditering med de nye krediteringsoplysninger 8. Systemet sender notifikation om afventede godkendelse af programmet med den opdaterede kreditering til systemadministrator <ul style="list-style-type: none"> 8.1. B13 Godkend program «extends»

Postkonditioner (Post conditions):
<ul style="list-style-type: none"> • Systemet har gemt kreditering med de nye krediteringsoplysninger fra Aktøren • Systemet afventer godkendelse af programmet med den opdaterede kreditering fra systemadministratoren
Alternative hændelsesforløb (Alternative flow):
<ul style="list-style-type: none"> • Aktør har mulighed for at afbryde brugsmønster fra punkt 2-6

Krydsreferencer:

Programmet med den opdaterede kreditering skal godkendes af systemadministrator i B13 Godkend program

Tabel 4.3 Detaljeret brugsmønster - opdater kreditering

Brugsmønster: Godkend program
ID: B13
Primære aktører: Systemadministrator
Sekundære aktører: Producer
Kort beskrivelse: Systemadministratoren skal have mulighed for at godkende et program som produceren har oprettet eller opdateret i systemet.
Prækonditioner (Pre-conditions):
<ul style="list-style-type: none"> • Systemadministratoren skal være logget ind • En producer har oprettet et program i systemet
Hovedhændelsesforløb (main flow):
<ol style="list-style-type: none"> 1. Brugsmønsteret starter når systemadministratoren vil godkende afventende programmer 2. Systemet viser en liste over afventende programmer 3. Systemadministrator vælger et afventende program 4. Systemet viser det valgte program og de tilhørende krediteringer <ol style="list-style-type: none"> 4.1. Systemadministratoren kan lave ændringer i det valgte program og dets krediteringer 4.2. Systemadministratoren gemmer eventuelle ændringer 5. Systemadministratoren godkender det valgte program med dertilhørende krediteringer 6. Systemet tilføjer programmet
Postkonditioner (Post conditions):
<ul style="list-style-type: none"> • Producer for pågældende program har fået notifikation om, at programmet er blevet godkendt • Systemet har tilføjet programmet

Alternative hændelsesforløb (Alternative flow):

- Ved punkt 2-4: Systemadministratoren kan vælge at afslutte brugsmønstret
- Hvis afsluttet under 4.1 eller 4.2 gemmes ændringer ikke
- Ved punkt 5: Systemadministratoren kan vælge at afvise programmet
 - Systemet tilføjer ikke programmet, og producenten bliver underrettet om afvisningen

Tabel 4.4 Detaljeret brugsmønster - Godekend program

KAPITEL 5

Analyse

I den statiske analysemodel fokuseres der på at identificere klasser og struktur, hvor identificeringen af klasser ofte sker som et led af en navneordsanalyse på de detaljeret brugsmønstre, og struktur igennem et analyseklassediagram. Navneordene fra navneordsanalysen beskriver mulige klassekandidater samt attributter, og analyseklassediagrammet beskriver relationerne mellem klasser og objekter [2].

I den dynamiske analysemodel (Adfærdsanalyse) udarbejdes brugsmønsterrealiseringen af relevante brugsmønstre. Ved brugsmønsterrealiseringen udarbejdes der et systeminteraktionsdiagrammer (system-sekvensdiagram), som har til formål at identificere system operationer, hvilket formelt bliver beskrevet i en systemoperationskontrakt. På baggrund deraf, bliver der udarbejdet et systemoperationssekvensdiagrammer (sekvensdiagram), som danner grundlag for et udvidet analyseklassediagrammet fra den statiske analysemodel, som indeholder relevante metoder [3].

5.1 Statisk analysemodel

Udarbejdelse af navneordsanalysen vil tage udgangspunkt i de detaljerede brugsmønstre fra Kapitel 4, B11 & B13. Tabel 5.1 & 5.2 repræsenterer navneordsanalysen, hvor alle navneord er markeret med rød.

Brugsmønster: Opdater kreditering
ID: B11
Primære aktører: Producer & Systemadministrator. (Omtales aktør fremover)
Sekundære aktører: Ingen
Kort beskrivelse: En producer skal kunne opdatere krediteringsoplysningerne for et program de ejer.

Prækonditioner (Pre-conditions):
<ul style="list-style-type: none"> • Programmet skal have en oprettet kreditering i systemet. • Aktøren har valgt et program.
Hovedhændelsesforløb (main flow):
<ol style="list-style-type: none"> 1. Brugsmønstret starter når en aktør ønsker at opdatere kreditering for et program i systemet 2. Aktøren vælger en kreditering fra programmet 3. Aktøren trykker på knappen opdater kreditering 4. Systemet viser den valgte kreditering for programmet 5. Aktør opdaterer felter med nye krediteringsoplysninger 6. Aktør trykker “opdater kreditering” 7. Systemet opdaterer den valgte kreditering med de nye krediteringsoplysninger 8. Systemet sender notifikation om afventede godkendelse af programmet med den opdaterede kreditering til systemadministrator <p>8.1. B13 Godkend program «extends»</p>
Postkonditioner (Post conditions):
<ul style="list-style-type: none"> • Systemet har gemt kreditering med de nye krediteringsoplysninger fra Aktør • Systemet afventer godkendelse af programmet med den opdaterede kreditering fra systemadministrator
Alternative hændelsesforløb (Alternative flow):
<ul style="list-style-type: none"> • Aktør har mulighed for at afbryde brugsmønster fra punkt 2-6
Krydsreferencer: Programmet med den opdaterede kreditering skal godkendes af systemadministrator i B13 Godkend program

Tabel 5.1 Navneordsanalyse af detaljeret brugsmønster - opdater kreditering

Brugsmønster: Godkend program
ID: B13
Primære aktører: Systemadministrator
Sekundære aktører: Producer
Kort beskrivelse: Systemadministratoren skal have mulighed for at godkende et program som Producenten har oprettet eller opdateret i systemet.

Prækonditioner (Pre-conditions): <ul style="list-style-type: none">• Systemadministratoren skal være logget ind• En Producer har oprettet et program i systemet
Hovedhændelsesforløb (main flow): <ol style="list-style-type: none">1. Brugsmønsteret starter når Systemadministratoren vil godkende afventende programmer2. Systemet viser en liste over afventende programmer3. Systemadministrator vælger et afventende program4. Systemet viser det valgte program og de tilhørende krediteringer<ol style="list-style-type: none">4.1. Systemadministratoren kan lave ændringer i det valgte program og dets krediteringer4.2. Systemadministratoren gemmer eventuelle ændringer5. Systemadministratoren godkender det pågældende program med dertilhørende krediteringer6. Systemet tilføjer programmet
Postkonditioner (Post conditions): <ul style="list-style-type: none">• Producer for pågældende program har fået notifikation om, at programmet er blevet godkendt• Systemet har tilføjet programmet
Alternative hændelsesforløb (Alternative flow): <ul style="list-style-type: none">• Ved punkt 2-4: Systemadministratoren kan vælge at afslutte brugsmønstret• Hvis afsluttet under 4.1 eller 4.2 gemmes ændringer ikke• Ved punkt 5: Systemadministratoren kan vælge at afvise programmet<ul style="list-style-type: none">– Systemet tilføjer ikke programmet, og produceren bliver underrettet om avisningen

Tabel 5.2 Navneordsanalyse af detaljeret brugsmønster - Godekend program

Med udgangspunkt i navneordsanalysen, vil alle markerede ord blive indsatt i Tabel 5.3, dog vil alle gentagelser blive frasorteret. Tabellen repræsenterer mulige klassekandidater og attributter.

Kreditering	Oplysning	Besked
Bruger	Databehandling	Oprettelse
Gæst	Forbindelse	Legitimationsoplysninger
Program	Eksportering	Konto
Skuespiller	Systemadministrator	Applikation
Producer	Dataanalyse	Loginvindue
Fotograf	Søgning	Brugernavn
Liste	Data	Adgangskode
Søgeresultater	Filformat	View
Resultat	Filplacering	Loginoplysninger
Brugsmønster	Markering	Pop-up-vindue
Aktør	Handling	Bekræftelse
Person	Værdi	Valg
Søgekategori	Information	Systemview
Serie	Programoplysninger	Godkendelse
Film	Krediteringsoplysninger	Notifikation
Krediteringsrolle	Forespørgsel	Mulighed
System	Knap	Ændring
Kald	Login	Afvisning
Database	Vindue	Opdateringsmuligheder
Søgekriterie	Felt	Systemlog
Svar	Status	Opdateringsdetaljer

Tabel 5.3 Klassenkandidater fundet ved navneordsanalyse

Tabel 5.4 indeholder en samlet oversigt over klasser og dertilhørende attributter, som er udarbejdet på baggrund af Tabel 5.3 og følgende regler:

- Fjernet synonymer
- Fjernet meta-navneord
- Fjernet navneord, der ikke kan defineres tydeligt
- Ændret navneord, der kan være tvetydige, men betyder det samme

Klasse	Attributter	Kommentarer
Kreditering (Credit)	creditedPerson role	Atributten "role" indeholder bl.a. ordene: "Skuespiller", "Fotograf" og "Producer"
Bruger (User)		
Gæst (Guest)		
Program (Programme)	name credits period eventId approved	Klassen er abstrakt. credits, period eventId er attributter gemt bag <i>programoplysninger</i> .

Person (CreditedPerson)		
Serie (TVSeries)	seasonMap episodeMap	seasonMap og episodeMap er skjulte attributter fra kontekst. Klassen arver fra Programme.
Udsendelse (Transmission)	programmeDetails	
Producer (Producer)		
Systemadministrator (SystemAdmin)	notifications	
Login (Login)	userMap	userMap har brugernavn og password (legitimationsoplysninger)
Systemlog (SystemLog)		
Handling (Action)		
Notifikation (Notification)		

Tabel 5.4 Klasser og attributter fundet gennem navneordsanalysen.

Herefter laves en definition af klasserne og der gives et eksempel på brugen af klassen, hvor det vurderes nødvendigt. Se Tabel 5.5.

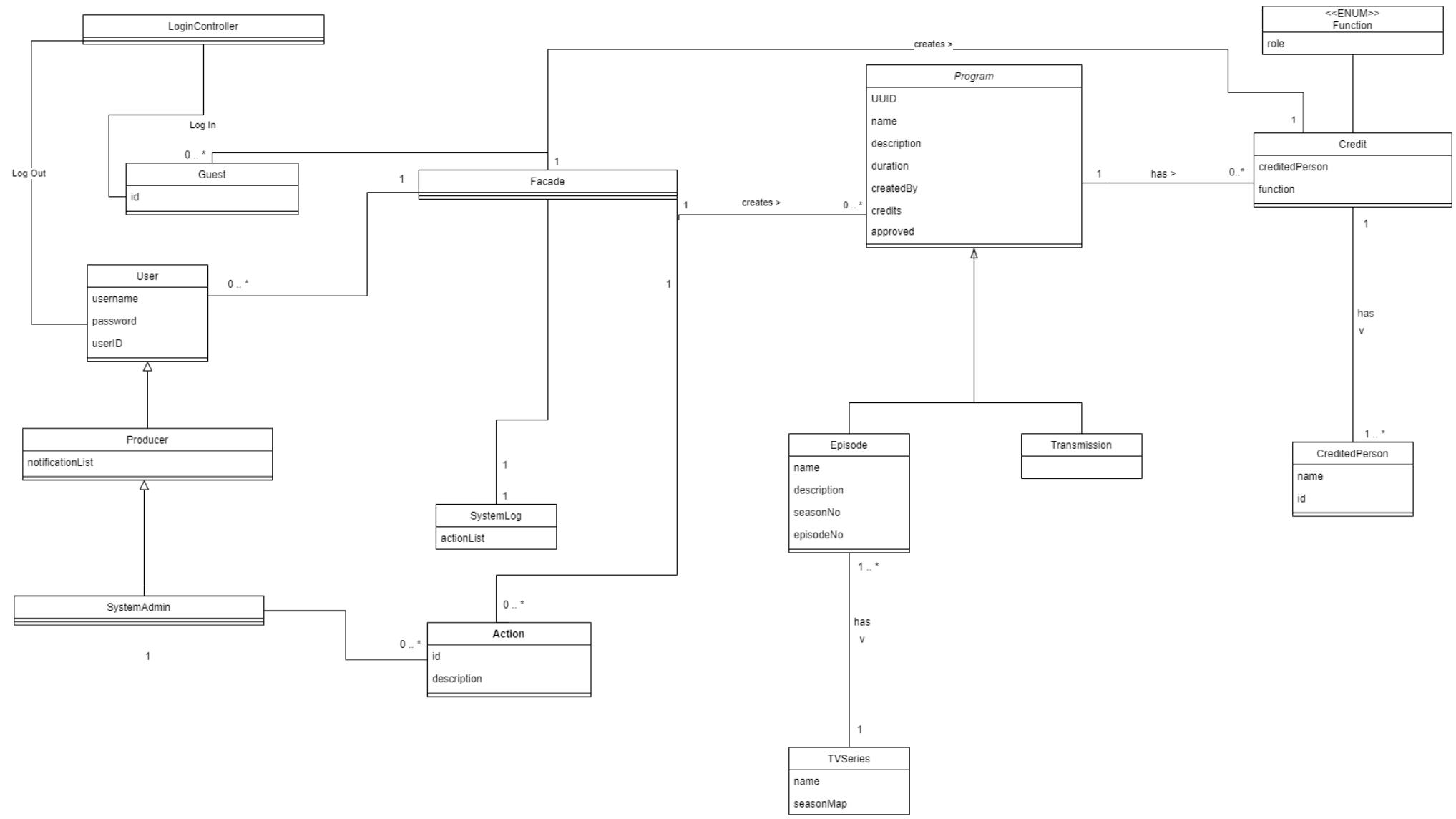
Klasse	Definition	Eksempel
Kreditering (Credit)	Credit-klassen indeholder én person og deres rolle.	Jens Hansen (creditedPersonId: 3) bliver krediteret som Billed- og lydredigeringsansvarlig (roleId: 2)
Bruger (User)	User-klassen har et brugernavn og password. Når gæsten logger in, vil denne blive tjekket med listen af Users i login-klassen. De kan også tilføje andre kontooplysninger som email. Desuden gemmes der en liste af de programmer, som brugeren har gemt.	En bruger med navnet Jens Hansen oprettet sig I systemet med brugernavnet jensHansen76 og password _jEnsHansen123. Han tilføjer sin email, og har en liste over programmer han har gemt.
Gæst (Guest)	Guest-klassen har applikationens mest begrænsede systemview, og har mulighed for både at oprette sig som en bruger samt at logge ind på en eksisterende bruger. Gæsten kan frit søge igennem applikationens programmer.	En Guest søger efter tv-serien <i>badehotellet</i> , for at finde den seneste episode, og gennemse krediteringerne.
Program (Programme)	Klassen er abstrakt og skal sikre polymorfi af forskellige typer af programmer (udsendelser og tv-serier). Et program har et navn.	
Person (CreditedPerson)	CreditedPerson-klassen skal indeholde et unikt ID for hver person, samt et navn.	En krediteret person ved navn Jens Hansen oprettes I systemet og bliver tildelt det unikke id “3”.

Serie (TVSeries)	TVSeries repræsentere serier fra programme-klassen, og indeholder information omkring og episoder, hvor episoder holder en referense til programoplysninger.	En serie fx "Bagedysten" oprettes I systemet med nestede maps der indeholder sæson/episodeMap par og episodeMap/programmeDetails par. Fx Bagedysten sæson 1-4, episode 1-10, med tilhørende programoplysninger til hver episode.
Udsendelse (Transmission)	En udsendelse består af alle programmer der ikke er serier, fx film, dokumentarer, sport. De har en attribut der indeholder programoplysningerne til udsendelsen.	Fx "Håndbold Finale", "Spiderman" og har programdetaljer til de tilhørende udsendelser.
Producer (Producer)	Producer-klassen har adgangsrettigheder, der gør den I stand til at oprette og opdatere programmer samt tilhørende krediteringer.	En given producer opretter programmet <i>Håndbold Finale</i> med relevante krediteringer
Systemadministrator (SystemAdmin)	Systemadministratoren skal have en liste over <i>Actions</i> , der skal godkendes. De har også adgang til systemloggen og får en notifikation, når der sker en ændring eller oprettelse af et program.	En Systemadministrator gennemlæser det nyoprettet program <i>Håndbold Finale</i> for fejl, og godkender programmet
Login (Login)	Login har et map, som holder styr på hvilke login der passer til hviken konto	
Systemlog (SystemLog)	Systemlog skal have en liste over <i>actions</i> foretaget I systemet.	Ny action, id: 5461318AcD, tidspunkt, 04/09-2025 17:15, brugertype, Producer
Handling (Action)	Action repræsentere hvilke handlinger der er foretaget I systemet af en producer eller systemadmin, og indeholder information over hvad der er foretager, og hvem som har foretaget handlingen samtidig med et unikt ID for handlingen	Producer, <i>Hans Hansen</i> , brugernavn, <i>HansHansenNordiskFilm</i> , ID: 17, har ændret programnavnet for program, <i>Chicago Fire</i> , til, <i>Chicago Med</i>
Notifikation (Notification)	Bliver oprettet når der sker en ændring i systemet lavet af en producer, som systemadministratoren skal være opmærksom på.	Bliver fx oprettet hvis en producer laver en ændring I en kreditering.

Tabel 5.5 Definition og eksempel af analyseklasserne

Med udgangspunkt i navneordsanalyse, er der udviklet et analyseklassediagram, Figur 5.1, som indeholder relationerne mellem klasserne og attributterne.

Domain



Figur 5.1 Analyseklassediagram - første udgave baseret på navneordsanalyse

5.2 Dynamisk analysemodel - Adfærdsanalyse

Ved udformningen af den dynamiske analysemodel laves der brugsmønsterrealiseringer af følgende brugsmønstre

- B05 Log-ind
- B06 Log-ud
- B10 Opret Program
- B11 Opdater kreditering
- B13 Godkend program
- B17 Opdater program

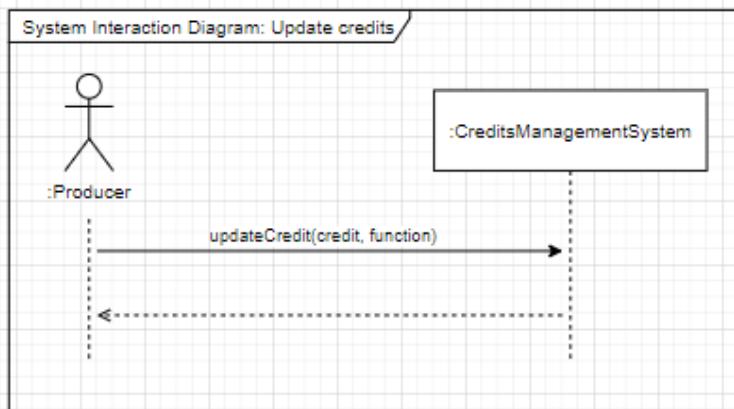
Afsnit 5.2 vil kun indeholde brugsmønsterrealiseringen af brugsmønster B11 *Opdater kreditering* og B13 *Godkend krediteringer*, hvor resten kan ses i Bilag C.

Brugsmønsterrealiseringssprocessen, som udarbejdes ud fra *happy day* senariet, begynder med opstillingen af systeminteraktionsdiagrammer, hvilket viser systemets eksterne systemoperationer, som aktørerne interagerer direkte med ud fra et *black box* syn på systemet. Systeminteraktionsdiagrammen leder til udarbejdelsen af en systemoperationskontrakt for hver systeminteraktionsdiagram, hvilke er formelle beskrivelser af systemoperationerne. En systemoperationskontrakt indeholder information om, hvilke ændringer, der forårsages af operationerne, samt hvilke ansvarsområder de har. Kontrakterne indeholder derudover krydsreferencer til deres respektive brugsmønstre samt prækonditioner og postkonditioner, hvilke beskriver systemets tilstand henholdsvis før og efter eksekveringen af operationen. På baggrund deraf udarbejdes der systemoperationssekvensdiagrammer, som er mere detaljerede sekvensdiagrammer, hvor der fokuseres på, hvad der sker i selve systemet igennem en række interne systemoperationer. Systemoperationssekvensdiagrammerne danner grundlag for udformningen af en revideret version af analyseklassediagrammet, hvor systemoperationerne, placeres på deres respektive klasse.

5.2.1 Brugsmønsterrealisering af B11 Opdater kreditering på program

På Figur 5.2 ses systeminteraktionsdiagrammet for brugsmønsteret B11. Diagrammet viser et 'black-box' view af aktørens interaktion med systemet igennem updateCredit

operationen. Operationen medtager selve krediteringen der skal opdateres, og den nye funktion, som krediteringen får tilskrevet sig, som metode parametre.



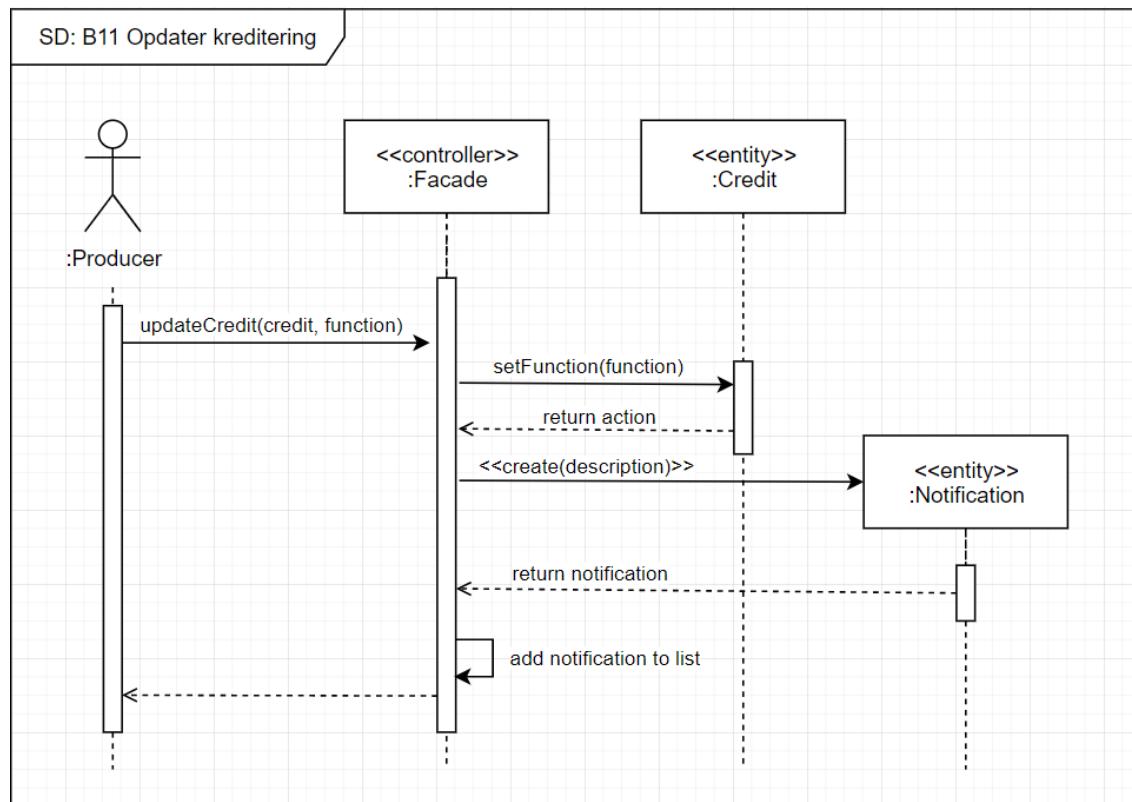
Figur 5.2 Systeminteraktionsdiagram - B11 Opdater kreditering på program

I Tabel 5.6 ses systemoperationskontrakten for brugsmønsteret B11. Kontrakten for brugsmønster B11 indeholder 3 ansvarsområder omhandlende adgangen til at tilgå det gældende programs krediteringer, at opdatere disse kreditering, samt at oprette en systemnotifikation, som sendes til systemadministratorens systemlog, hvilket også angives ved kontraktens output. Prækonditionen for operationen er en allerede eksisterende kreditering, og postkonditionen er, at den opdaterede kreditering gemmes med den nye information.

Contract	
Operation:	updateCredit(credit, function)
Cross reference:	B11 Opdater kreditering på program
Responsibility	<ul style="list-style-type: none"> • At tilgå et programs eksisterende krediteringer. • At opdatere hvilken funktion en person har i det givne program. • At oprette en notifikation.
Output	<ul style="list-style-type: none"> • En notifikation.
Preconditions	Programmet skal have en oprettet kreditering i systemet.
Postconditions	Systemet har gemt kreditering med den nye information fra Aktør.

Tabel 5.6 Systemoperationskontrakt - B11 Opdater kreditering på program

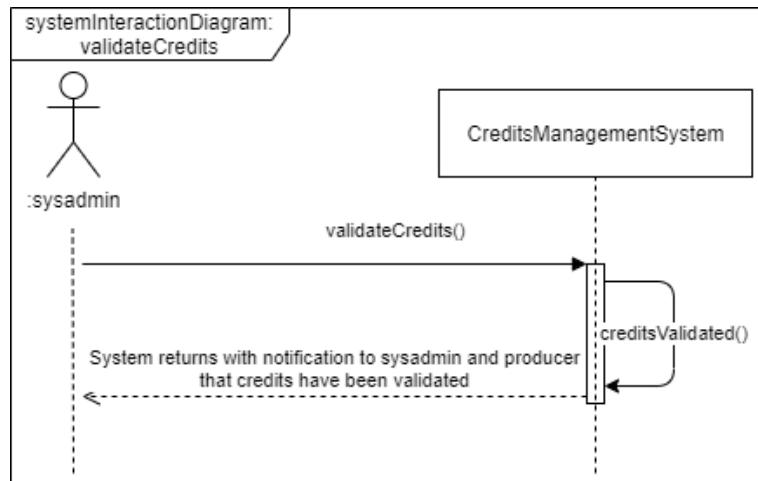
På Figur 5.3 ses systemoperationssekvensdiagrammet for brugsmønsteret B11. Heri ses, hvordan den eksterne operationen, *updateCredit()*, starter en serie af interne operationer i systemet, som returnerer en opdateret kreditering samt en oprettet notifikation. Sekvensdiagrammet viser, hvordan systemet ændrer den gældende krediterings funktion til den medsendte, hvorefter notifikationen med de gældende ændringer tilføjes til systemets liste af notifikationer.



Figur 5.3 Systemoperationssekvensdiagram - B11 Opdater kreditering på program

5.2.2 Brugsmønsterrealisering af B13 Godkend krediteringer

På Figur 5.4 ses systeminteraktionsdiagrammet for brugsmønster B13. Diagrammet viser, hvordan aktøren, som i dette brugsmønster er systemadministratoren, interagere med den eksterne operation *approveProgram*. Operationen har til formål at godkende krediteringer på et givet program og herefter at returnere en notifikation til systemadmin og producer.



Figur 5.4 Systeminteraktionsdiagram - B13 Godkend krediteringer

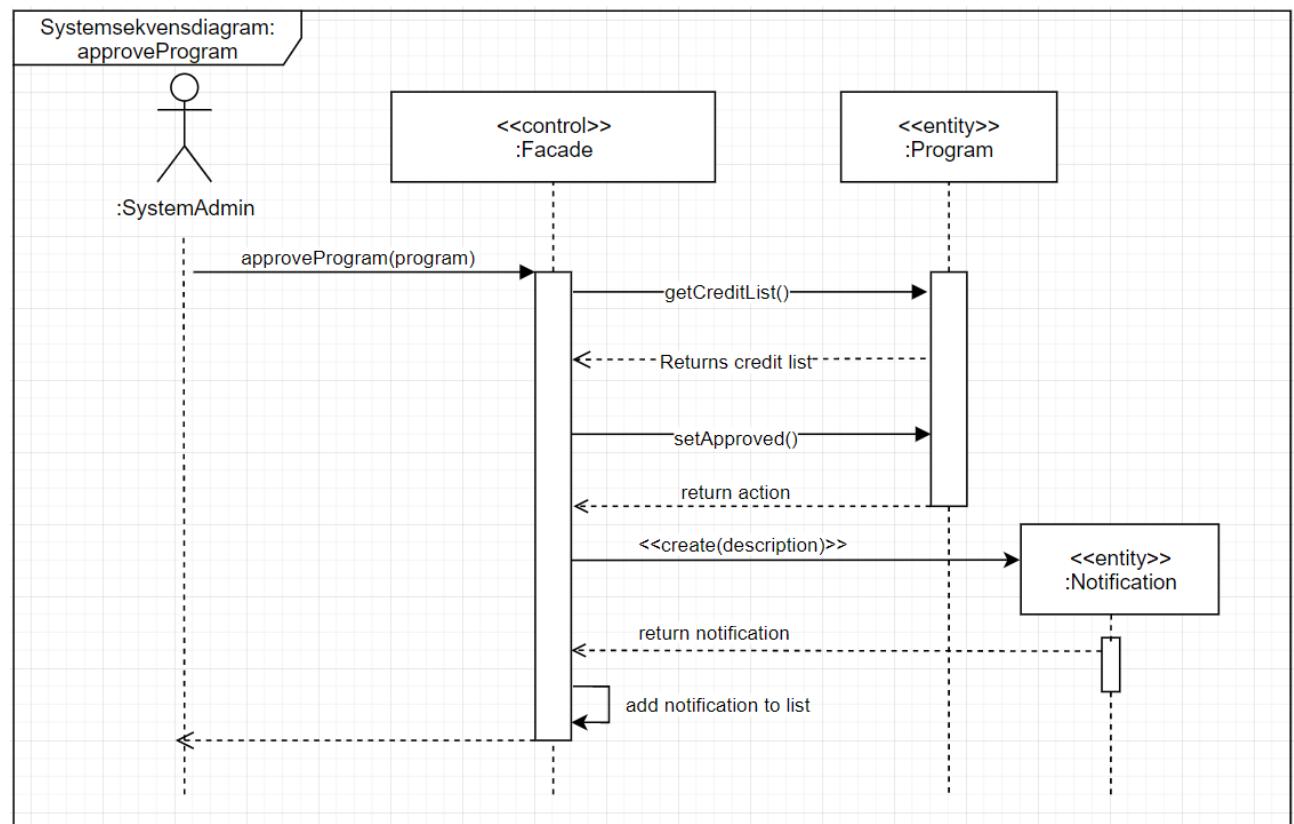
I Tabel 5.7 ses systemoperationskontrakten for brugsmønster B13. Kontrakten indeholder to ansvarsområder, at kunne godkende krediteringer på et program samt at oprette en systemnotifikation, hvilket er operationens output. Prækondition for operationen er, at brugeren er logget ind som systemadministrator, samt at en producer i systemet har oprettet mindst én kreditering, som skal godkendes. Postkonditionen kræver at de gældende krediteringer på et given program er blevet godkendt, og at notifikationen er blevet oprettet.

Contract	
Operation:	approveProgram(program)
Cross reference:	B13 Godkend krediteringer
Responsibility	<ul style="list-style-type: none"> • At validere krediteringer lavet af en producer • At lave en notifikation
Output	<ul style="list-style-type: none"> • En notifikation.
Preconditions	<ul style="list-style-type: none"> • System administrator er logget ind • En producer har lavet en kreditering på et program

Postconditions	<ul style="list-style-type: none"> De oprettede krediteringer er blevet godkendt af systemadministrator og en notifikation er blevet oprettet.
-----------------------	---

Tabel 5.7 Systemoperationskontrakt - B13 Godkend krediteringer

På Figur 5.5 ses systemoperationssekvensdiagrammet for brugsmønster B13. Diagrammet viser den serie af interne operationer som igangsættes, når systemadministratoren interagerer med operationen approveProgram(). Operationen tager det gældende program som parameter i metoden, hvorefter Facaden henter de gældende krediteringer på programmet. Dernæst godkendes krediteringerne og programmet. Endeligt oprettes en tilhørende notifikation med de relevante informationer for godkendelsen.

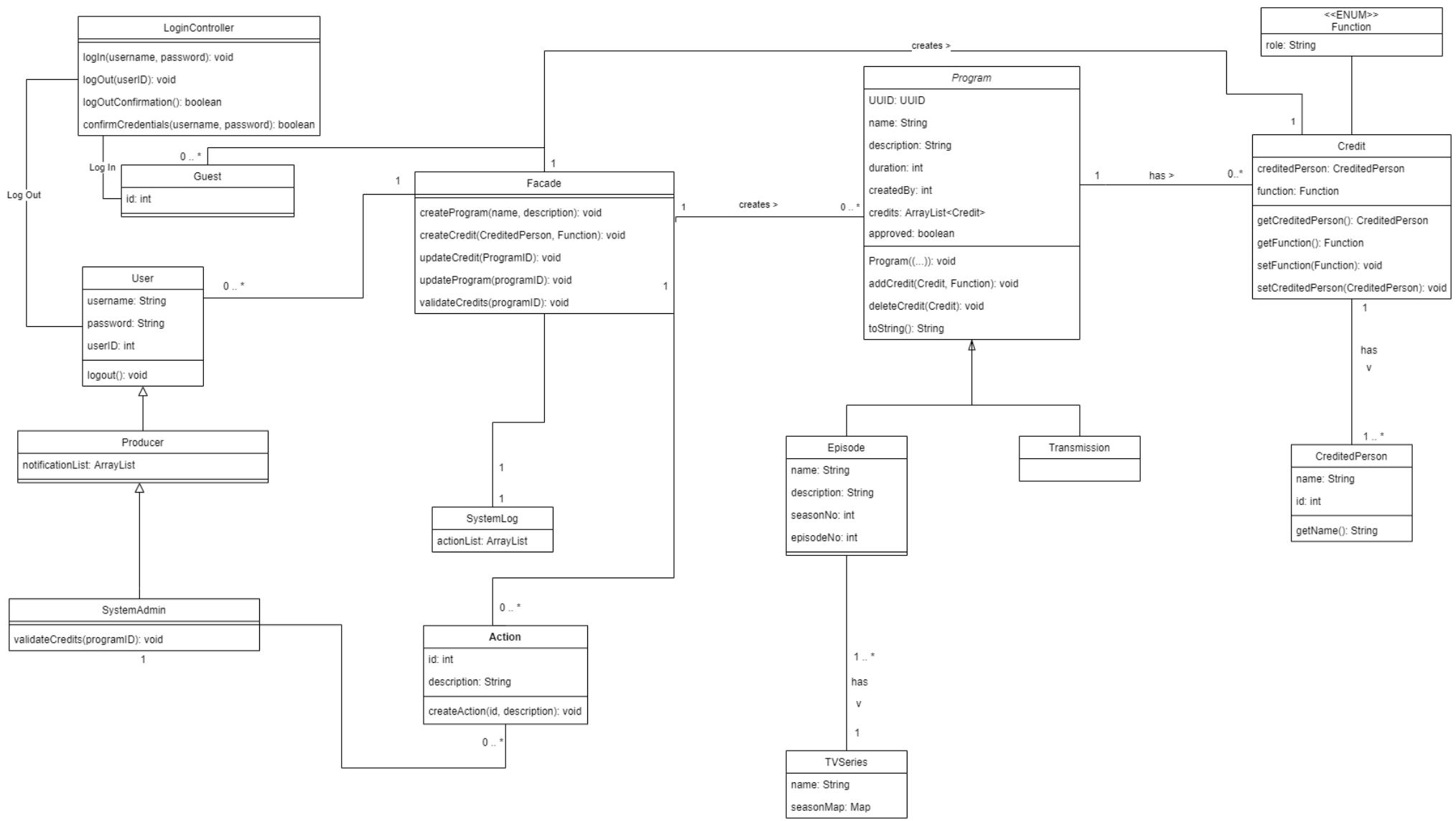


Figur 5.5 Systemoperationssekvensdiagram - B13 Godkend krediteringer

5.2.3 Analyseklassediagram

Baseret på analyseklassediagrammet, Figur 5.1, fra Kapitel 5.1, er der udarbejdet et opdateret analyseklassediagram, som indeholder metoderne fundet gennem brugsmønsterrealiseringen. Det opdaterede analyseklassediagram kan ses på Figur 5.6

Domain



Figur 5.6 Analyseklassediagram af domæne-laget - Yderligere defineret i forhold til Analyseklassediagrammet Figur 5.1

KAPITEL 6

Design

For design af Credits Management Systemet til TV2, har gruppen foretaget bestemte design-valg. Designvalgene tager fodfæste fra det tidligere udarbejdet analyseklassediagram fra Kapitel 5 og designklassediagram fra samme kapitel. Designvalgene afspejler hvordan Java-koden i programmet er designet og opbygget. Der er også blevet udarbejdet et databasedesigndiagram som beskriver fundamentet for selve databasen

6.1 Design af Credits Management System

Projektet er udarbejdet gennem processen UP, hvilket, for implementeringen, betyder, at der hurtigt kan ske ændringer på de definerede krav. Derfor er det essentielt, at koden kan ændres hurtigt og nemt, uden for meget andet kode bliver påvirket. På baggrund deraf, er det valgt, at programmet designes efter en tre lags arkitektur. Tre lags arkitekturmødellen, som består af et præsentations- domæne- og persistenslag, sikrer nemlig, at ændringer eksempelvis på persistenslaget ikke direkte påvirker præsentationslaget, hvilket gør at ændringer i koden, som udgangspunkt, kan ske hurtigt og nemt, hvis der kommer nye eller ændringer af krav.^[4] Tre lags arkitekturen vil samtidig være med til at overskueliggøre kommunikationen internt i programmet, og ville give en bedre forståelse for, hvordan et sådant program kunne udarbejdes i industrien^[5].

Den valgte arkitektur er afspejlet i designklassediagrammet på figur 6.1 ved opdeling i pakkerne *Presentation*, *Domain* og *Persistence*.

Præsentations-laget håndterer Graphical User Interface og består af forskellige controllers for hver type af bruger af systemet herunder *Guest*-, *Producer*- og *System administrator controller*. På dette lag findes også en controller til at håndtere login i systemet.

Domæne-laget indeholder programlogikken som håndterer input fra brugeren, og admi-

nistrerer de handlinger der skal til for at programmet kan køre efter hensigten. Domæne-laget indeholder endvidere klassen *Facade* som fungerer som kommunikationscentral mellem lagene.

Persistens-laget håndterer oprettelse af kommunikation til databasen samt eksekvering af SQL-queuries. I dette lag er der også en klasse der håndterer export af data til en fil samt en config-klasse for databaseopsætning for PostgreSQL.

Der er som udgangspunkt kun kommunikation gennem lagene i en nedadgående retning i den traditionelle måde at opstille den lagdelte arkitektur på. Der er dog valgt i dette projekt at benytte sig af *dependency inversion*, da det viste sig nyttigt, at et lavere liggende lag kunne bruge funktionaliteten fra et højere liggende lag.^[6] Dermed er der implementeret et interface for *PersistenceHandler* liggende i domænelaget for at overholde denne måde at opstille lagene på.

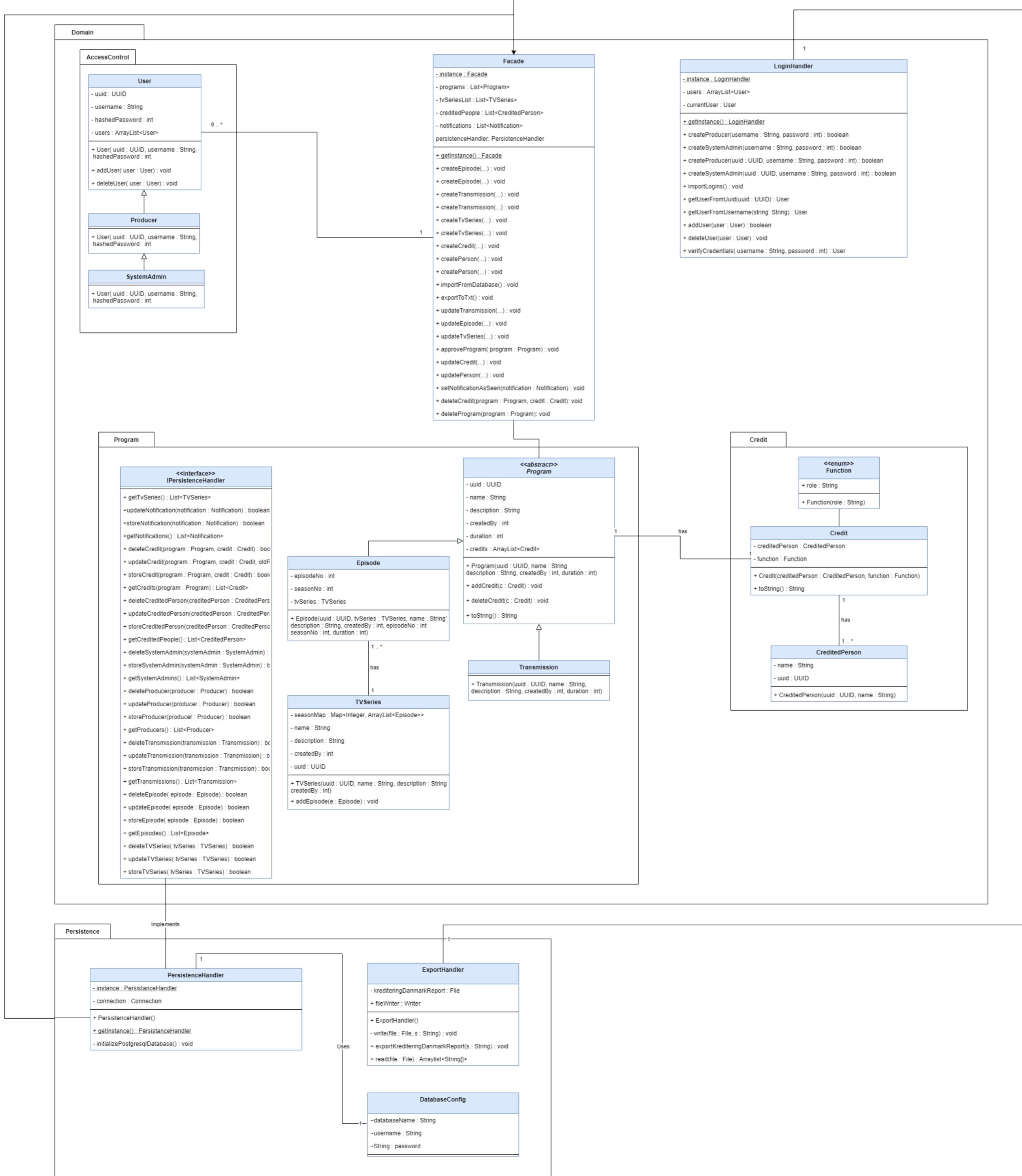
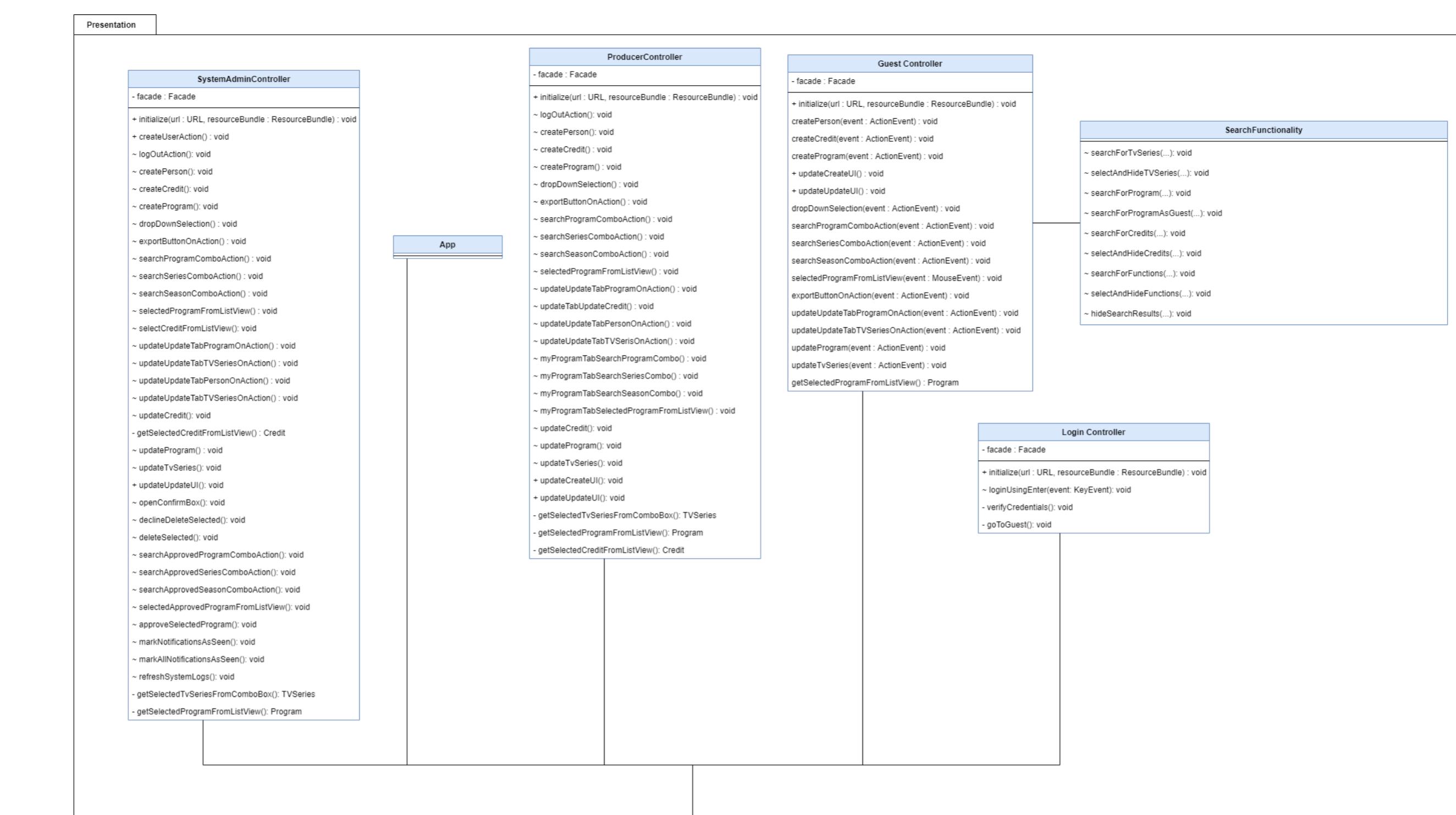
På designklassediagrammet findes der, foruden pakkeopdeling af lagene, yderligere pakkeopdeling af klasser der er tæt forbundet i programmet. Herunder ses pakkerne *AccessControl*, *Program* og *Credit*. Disse pakker indeholder klasser tæt relateret til hinanden for at overskueliggøre designet af programmet.

Designklassediagrammet indeholder også access modifiers som er sat som default med ~ tilde. Det er standard for FXML-funktioner genereret igennem SceneBuilder, at disse har default som access-modifier og derfor er disse ikke ændret.

Det er også bemærkelsesværdigt, at der er klasser med metoder, hvor parameterlisterne er set bort fra. Dette er gældende for eksempelvis metoder på klassen *Facade*, da den indeholder mange metoder med lange - og meget ens, parameterlister. Disse er set bort fra for læsbarhedens skyld med henblik på, at UML-diagrammet er til internt brug i gruppen.^[7] Et komplet designklassediagram, genereret ved brug af IntelliJ, kan findes i bilag H.

6.2 Design af database

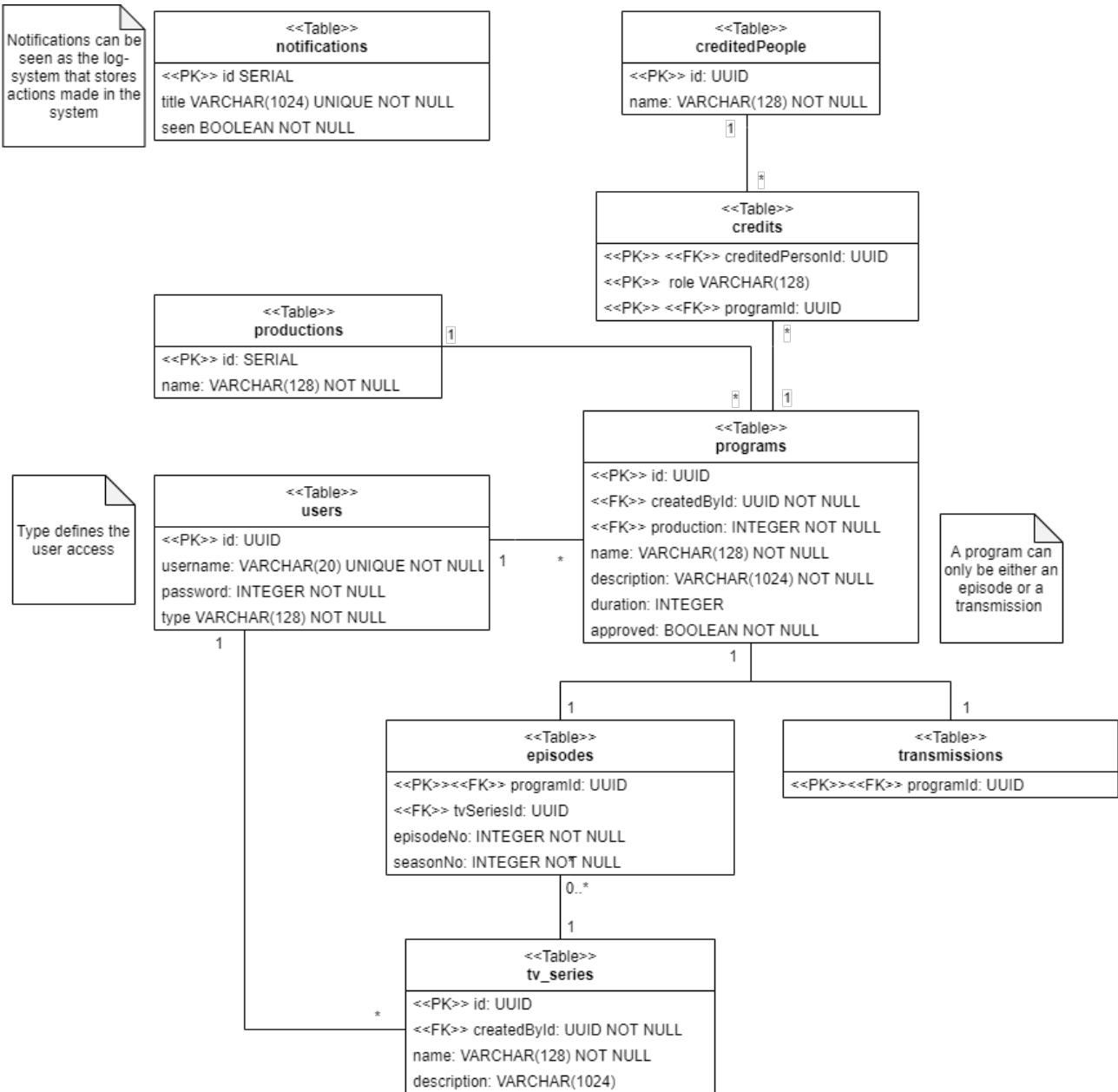
En del af problemformuleringens underspørgsmål, Kapitel 1.3, er, at opsætte en database for et Credits Management System, hvor der, på baggrund af forståelsen for systemet, baseret på designklassediagrammet, Figur 6.1, samt resultaterne fra analyseafsnittet, Afsnit 5, er der udarbejdet et design diagram for, hvordan databasen skal se ud. Diagrammet er udarbejdet med brug af UML-notioner og betragtes i normalform fire.



Figur 6.1 Designklassediagram

Alle de tabeller i databasen som svarer til en klasse med et UUID i Java, bruger et UUID som primary key. Her kan der gøres brug af datatypen UUID, som er indbygget i PostgreSQL. Den mest centrale del af databasen er "programs"-tabellen, som indeholder informationen fra *Program*-klassen i Java. Et program kan enten være en Transmission eller Episode, hvilket er grunden til "episodes" og "transmissions"-tabellerne. De benytter sig af foreign key inheritance - dvs. episodes og transmissions er en specialisering af programs, ligesom i Javaprogrammet. Der gøres brug af foreign key inheritance fremfor PostgreSQL's inheritance, da PostgreSQL's inheritance ikke opretter en række i parent tabellen, og derfor ville programId i "credits"-tabellen ikke kunne have en foreign key constraint ved brug af dette. Ved at bruge foreign key inheritance, placeres de attributter som episodes og transmissions har tilfælles i programs tabellen, og de specialiserede attributter i deres respektive tabeller.

Hver række i episodes eller transmissions svarer til én række i programs. Episodes refererer desuden tv_series, og både programs og tv_series refererer users gennem deres "createdById"-attribut. Programs refererer productions. Credits referer creditedPeople tabellen gennem creditedPersonId attributten og programs gennem programId attributten. Credits tabellen har en composite key som primary key bestående af creditedPersonId, role og programId, da én person per definition kun kan være krediteret med den samme rolle én gang ved hvert program. Notifications står for sig selv uden nogle referencer, og indeholder systemets "handlings-log". Se Figur 6.2.



Figur 6.2 Database Design diagram

KAPITEL 7

Implementering

Tages et blik tilbage på problemformuleringen, Kapitel 1.3, blev spørgsmålet, hvorvidt man kunne lave en brugergrænseflade til et Credits Management System, stillet. Systemet består af en GUI, der er designet og opsat gennem editoren Scenebuilder. Igennem nævnte editor outputtes brugergrænsefladen som FXML-filer, hvilke bruges i en kombination med relevante controllers, for at skabe brugergræsnefladen. Baggrunden for valget af Scenebuilder er, at det fjerner en del manuelt arbejde, i forhold til hvis gruppen skulle kode alle grafiske elementer i FXML-filer, hvilket gør, at der kan bruges mere tid på reel logik. Der vil blive set nærmere på den konkrete implementering af brugsmønstrene B11 og B13 samt implementering af databasen:

- B11 Opdater kreditering på program
 - Rettighedstjek
 - Notifikationsfunktionalitet
- B13 Godkend krediteringer
- SQL-Implementering

7.1 Implementering af B11 Opdater kreditering på program

For at kunne opdatere en kreditering, skal der vælges en specifik kreditering på et specifikt program igennem GUI'en, hvorefter knappen *Update credit* vil blive tilgængelig. Tilgængeligheden afhænger dog også af, at det skal være samme producer, som oprettede selve programmet, som er logget ind i systemet, eller hvis brugeren har administrator rettigheder. Brugeren bliver ført til en ny tab, *Update*, og kan der ændre på den givne kreditering.

```
546     @FXML
547     void updateCredit() {
548         try {
549             //To get the credit based on the selected "item" from
550             //the credit list view
551             Credit credit = getSelectedCreditFromListView();
552             Credit.Function function = Facade.getInstance().
553                 getFunctions().get(functionUpdateSelection.
554                 getSelectionModel().getSelectedIndex());
555
556             Facade.getInstance().updateCredit(credit, function);
557         } catch (IndexOutOfBoundsException e) {
558         }
559     }
```

Listing 7.1: Kode udklip af updateCredit-metoden fra controllerklassen producer

Metoden updateCredit, Listing 7.1, som ligger i systemAdmin- & producercontrolleren, bliver kaldt fra knappen *Update credit* ved tabben *Update* i GUI'en. Igennem listen med krediteringer fra tabben *Search/View* og/eller *My Programs*, bliver instansen af den givne kreditering hentet og sat lig med en lokal variabel (local variable) kaldet *credit*, Linje 550, som bliver parset ind i updateCredit-metoden ved facade-klassen, Linje 553, sammen med den opdateret funktion, rolle, som krediteringen skal ændres til, hvilken hentes egennem comboboksen *functionUpdateSelection* fra *update* tabben, Linje 551.

```
398     public void updateCredit(Credit credit, Credit.Function
399         function) {
400         String oldRole = credit.getFunction().role;
401         credit.setFunction(function);
402         Program program = getProgramFromCredit(credit);
403         if (program.isApproved()){
404             program.setApproved(false);
405         }
406         persistenceHandler.updateCredit(getProgramFromCredit(
407             credit), credit, oldRole);
408         Notification notification = new Notification(String.format
409             ("%s updated a credit for %s on %s from %s to now be %s
410             on %s",
411             LoginHandler.getInstance().getCurrentUser().
412                 getUsername(), credit.getCreditedPerson().
413                     getName(),
414             getProgramFromCredit(credit).getName(), oldRole,
```

```
        function.role, currentTime())));
409    notifications.add(notification);
410    persistenceHandler.storeNotification(notification);
411 }
```

Listing 7.2: Kode udklip af updateCredit-metoden fra facade-klassen

Metoden updateCredit, Listing 7.2 ved facadeklassen tager de to parametre parset med fra metoden i controlleren, og på baggrund deraf opdaterer den medsendte kreditering til dens nye funktion, rolle, Linje 401. Hvis den opdaterede kreditering ligger på et program, som allerede er godkendt, sættes programmet til at skulle godkendes endnu engang, Linje 402, hvilket er med til at sikre, at ændringen ikke indeholder fejl. Den opdaterede kreditering gemmes i databasen, Linje 405, ved at kalde updateCredit metoden i PersistenceHandler-klassen. Endeligt, for at logge aktiviteten i systemet, oprettes en ny instans af notifikation, Linje 406, som indeholder oplysninger omkring, hvem der udførte handlingen, hvad der blev ændret og tidspunktet for ændringen. Den nye notifikation gemmes naturligvis også i databasen, Linje 410, igennem storeNotification metoden i PersistenceHandler-klassen.

```
548     @Override
549     public boolean updateCredit(Program program, Credit credit,
550         String oldRole) {
551         try {
552             PreparedStatement stmt = connection.prepareStatement(
553                 "UPDATE credits SET role = ? WHERE
554                 creditedPersonId = ? AND programid = ? AND
555                 role = ?");
556             stmt.setString(1, credit.getFunction().role);
557             stmt.setObject(2, credit.getCreditedPerson().getUuid());
558             stmt.setObject(3, program.getUuid());
559             stmt.setString(4, oldRole);
560
561             PreparedStatement stmt1 = connection.prepareStatement(
562                 "UPDATE programs SET approved = false WHERE id
563                 = ?");
564             stmt1.setObject(1, program.getUuid());
565
566             stmt.execute();
567             stmt1.execute();
568             return true;
569         } catch (SQLException e) {
```

```
566         e.printStackTrace();
567         return false;
568     }
569 }
```

Listing 7.3: Kode udklip af updateCredit-metoden fra persistenceHandler-klassen

De nye ændringer gemmes i databasen, hvilket gøres gennem updateCredit metoden i PersistenceHandler-klassen, Listing 7.3. Selve metoden opretter et PreparedStatement-objekt til at eksekvere et SQL-statement, Linje 551 & Linje 558, der opdaterer de relevante celler i den relevante tabel igennem de parametre som metoden tager med. Metoden har to PreparedStatement-objekter, hvilket skyldes at der skal opdateres data i to forskellige tabeller. Det første PreparedStatement-objekt, stmt Linje 551, sørger for at opdatere selve krediteringen hvor stmt1, Linje 558, sørger for, at programmet, også i databasen, bliver sat til at skulle godkendes. Der er gjort brug af spørgsmålstegegnede steder, hvor der skal indsættes en parameter, for at undgå SQL-injection. SQL-injection er, kort sagt, når systemet prompter om et brugerinput, hvortil brugeren giver en SQL-statement, som ubevidst bliver kørt i databasen[8]. For at indsætte værdier hvor spørgsmålstegegne er placeret, gøres der brug af set-metoder, Linje 553-556 for stmt & Linje 560 for stmt1, som i dette tilfælde har datatypen String (setString) og Object (setObject), der kan tilgås igennem PreparedStatement-objektet.

7.2 Implementering af B13 Godkend krediteringer

For at et program bliver vist *officielt* i systemet, skal systemadministratoren godkende et program og de tilhørende krediteringer, hvilket funktionelt gøres inde på *Approve*-tabben. På denne tab er der et listview, som indeholder alle de nyoprettede eller opdaterede programmer, som afventer godkendelse. Systemadministratoren kan her godkende et program og dets krediteringer ved at trykke på programmet, hvor krediteringerne vil blive vist i et andet listview, og derefter på godkend knappen, som kalder *approveSelectedProgram*-metoden i SystemAdminContoller. Se Listing 7.4. Denne metode kalder *approveProgram*-metoden fra *Facade-klassen*, Listing 7.5, som tager et program som metodeparameter. Programmet fås igennem et kald til *getSelectedProgramFromApprovedListView*-metoden, Linje 665, som tager værdien fra den valgte række i listviewet, som er en værdi af typen String. Stringen indeholder både programmets navn samt et unikt id adskilt af et kolon. Stringen splittes i to ved kolonet, hvorefter metoden returnerer det gældende program baseret på det unikke id¹.

¹Funktionaliteten for *getSelectedProgramFromApprovedListView*-metoden er kun beskrevet, ikke medtaget som kodeudklip

```
663     @FXML
664     void approveSelectedProgram() {
665         Facade.getInstance().approveProgram(
666             getSelectedProgramFromApprovedListView());
667     }
```

Listing 7.4: Kode udklip af approveSelectedProgram-metoden fra controllerklassen SystemAdmin

Metoden *approveProgram*, 7.5 starter med at ændre det gældende programs approved-attribut, Linje 386, som har datatypen boolean, til true, hvilket repræsenterer, at programmet er godkendt. Det forløber sig igennem *Program-klassens* setter-metode for approved-attributten kaldet *setApproved*. Som et led i metoden, opdateres databasens instans af programmet ved enten at kalde metoden *updateEpisode*, hvis programmet er en episode, eller *updateTransmission*, hvis programmet er en transmission, i *PersistenceHandler-klassen*, Linje 387-391. Metoderne *updateEpisode* og *updateTransmission* fungerer til dels på samme måde som metoden *updateCredit* fra *PersistenceHandler-klassen* beskrevet i Listing 7.3. Her er det istedet programs-, episodes- og transmissions-tabellerne i databasen der opdateres baseret på typen af det godkendte program. Til sidst i metoden oprettes der en ny instans af *Notification-klassen*, Linje 392, som indeholder relevante informationer om den foretagede ændring ligesom beskrevet under Listing 7.2, og som naturligvis også gemmes i databasen, Linje 395.

```
385     public void approveProgram(Program program) {
386         program.setApproved(true);
387         if (program instanceof Episode) {
388             persistenceHandler.updateEpisode((Episode) program);
389         } else if (program instanceof Transmission) {
390             persistenceHandler.updateTransmission((Transmission)
391                 program);
392         }
393         Notification notification = new Notification(String.format
394             ("%s approved a program with the UUID %s and name %s on
395             %s",
396             LoginHandler.getInstance().getCurrentUser().
397                 getUsername(), program.getUuid().toString(),
398                 program.getName(), currentTime()));
399         notifications.add(notification);
400         persistenceHandler.storeNotification(notification);
401     }
```

Listing 7.5: Kode udklip af approveProgram-metoden fra Facade-klassen

7.3 SQL-Implementering

Systemet gør brug af en database for persistering af data, men grundet gruppens faglige stadie, kører databasen kun lokalt. Det skaber også muligheden for, at alle gruppemedlemmer nemt kan arbejde på systemet samtidig, og foretage ændringer der kan påvirke databasen, uden at skabe *real time* konflikter, som ligger væk fra gruppens faglige stadie. Ved Listing 7.6, som viser database-scriptet, ses, at alle tabeller bliver droppet, slettet, hvergang scriptet køres², Linje 1-11, hvilket er med til at sikre, at alle gruppemedlemmer har samme database, som kan arbejdes ud fra når scriptet blev kørt. Det var hensigten, at der ville blive gjort brug af følgende statement `DROP DATABASE IF EXISTS database_name ;`[9]. Desværre blev databasen ikke droppet, hvorfor hver tabel istedet er blevet droppet individuelt. Selve scriptet indeholder også dummy-data, hvilket er til, for at give systemet noget indhold første gang det køres. Se Bilag D for det fulde SQL-Script.

Ved Listing 7.6 ses brugen af en såkaldt transaktion, som repræsenteres af en række keywords. Transaktionen starter ved keywordet `BEGIN;`, Linje 13, og slutter ved keywordet `COMMIT;`, Linje 72, hvilket betyder, at transaktionen omringer oprettelsen af databasens tabeller. Brugen af transaktioner introducerer begrebet atomicitet, som betyder, at alle operationerne, der er i transaktionen, skal gennemføres før ændringerne træder i kraft[10]. Dette betyder altså, at alle databasens tabeller kun oprettes, hvis ingen af de individuelle tabeloprettelser støder på problemer. Endnu en transaktion benyttes til at indsætte data i databasen, hvilket ses i det fulde database-script i Bilag D Listing 1. Her starter transaktionen på linje 82 og slutter på linje 160. Denne transaktion er særlig vigtig, da scriptets `INSERT`-statements indsætter en række værdier, som danner krydsreferencer mellem tabellerne i form af foreign keys. Dette kræver en vis konsistens mellem de krydsrefererende værdier, da der ellers risikeres fejl, når der hentes data fra databasen.

```
1  DROP TABLE IF EXISTS programs CASCADE;
2  DROP TABLE IF EXISTS productions CASCADE;
3  DROP TABLE IF EXISTS users CASCADE;
4  DROP TABLE IF EXISTS transmissions CASCADE;
5  DROP TABLE IF EXISTS episodes CASCADE;
6  DROP TABLE IF EXISTS credits CASCADE;
7  DROP TABLE IF EXISTS tv_series CASCADE;
8  DROP TABLE IF EXISTS admins CASCADE;
9  DROP TABLE IF EXISTS producers CASCADE;
10 DROP TABLE IF EXISTS creditedPeople CASCADE;
```

²Antaget at filen hver gang bliver kørt fra top til bund

```
11  DROP TABLE IF EXISTS notifications CASCADE;
12
13 BEGIN;
14 CREATE TABLE creditedPeople(
15     id UUID PRIMARY KEY,
16     name VARCHAR(128) NOT NULL
17 );
18
19 CREATE TABLE productions(
20     id SERIAL PRIMARY KEY,
21     name VARCHAR(128) NOT NULL
22 );
23
24 CREATE TABLE users(
25     id UUID PRIMARY KEY,
26     username VARCHAR(20) NOT NULL,
27     password INTEGER NOT NULL,
28     type VARCHAR(128) NOT NULL
29 );
30
31 CREATE TABLE programs(
32     id UUID PRIMARY KEY,
33     name VARCHAR(128) NOT NULL,
34     description VARCHAR(1024) NOT NULL,
35     createdById UUID NOT NULL REFERENCES users(id),
36     duration INTEGER NOT NULL,
37     approved BOOLEAN NOT NULL,
38     production INTEGER NOT NULL REFERENCES productions(id)
39 );
40
41 CREATE TABLE transmissions(
42     programsId UUID PRIMARY KEY REFERENCES programs(id)
43 );
44
45 CREATE TABLE episodes(
46     programsId UUID PRIMARY KEY REFERENCES programs(id),
47     tvSeriesId UUID NOT NULL,
48     episodeNo INTEGER NOT NULL,
49     seasonNo INTEGER NOT NULL
50 );
51
```

```
52 CREATE TABLE credits(
53     creditedPersonId UUID NOT NULL REFERENCES creditedPeople(id),
54     role VARCHAR(128) NOT NULL,
55     programId UUID NOT NULL,
56     PRIMARY KEY (creditedPersonId, role, programId)
57 );
58
59 CREATE TABLE tv_series(
60     id UUID PRIMARY KEY,
61     name VARCHAR(128) NOT NULL,
62     description VARCHAR(1024),
63     createdByUuid UUID NOT NULL REFERENCES users(id)
64 );
65
66 CREATE TABLE notifications(
67     id SERIAL PRIMARY KEY,
68     title VARCHAR(1024) UNIQUE NOT NULL,
69     seen BOOLEAN NOT NULL
70 );
71
72 COMMIT;
73
74 CREATE VIEW transmissionsView AS
75 SELECT id, name, description, createdByUuid, duration, approved,
    production FROM programs, transmissions
76 WHERE programs.id = transmissions.programsId;
77
78 CREATE VIEW episodesView AS
79 SELECT id, tvSeriesId, name, description, createdByUuid, episodeNo,
    seasonNo, duration, approved, production FROM programs,
    episodes
80 WHERE programs.id = episodes.programsId;
```

Listing 7.6: Systemets SQL-Script

KAPITEL 8

Test

På baggrund af implementeringen findes det essentielt at udarbejde en undersøgelse, som viser hvordan den egentlige implementering passer på de udarbejdet brugsmønstre fra Kapitel 5, samt hvorvidt implementeringen kan betragtes som en succes i henhold til problemformuleringen. Undersøgelsen dækker over udførelsen af unittest & brugsmønsterstest. Endvidere har undersøgelsen til formål at give gruppen feedback på, hvorledes funktionaliteten ved implementeringen og den opstillet GUI, kan vurderes tilfredsstillende, hvilket undersøges på baggrund af en brugertest.

8.1 Unittest

Gruppen har lavet automatiseret white box test af de mest centrale enheder i programmet ved brug af *defacto Java frameworket JUnit 4.12*. Testene laves for at verificere at klasser og metoder virker korrekt. Desuden bruges Maven for at køre klasserne.

Der testes 3 metoder i facadeklassen. For at teste denne klasse, skal den først sættes op. Dette gøres i `setUp()` metoden, Linje 20 til 25, som kører før testmetoderne. Se Listing 8.1. Her hentes instanserne fra Singleton klasserne og data fra databasen importeres, sådan at programmet fungerer og kan testes.

```
15 public class FacadeTest {
16     Facade facade;
17     LoginHandler loginHandler;
18
19     @Before
20     public void setUp() {
21         facade = Facade.getInstance();
22         facade.importFromDatabase();
23         loginHandler = LoginHandler.getInstance();
24         loginHandler.importLogins();
```

```
25     loginHandler.setCurrentUser(loginHandler.getUsers().get(0)
26         );
27 }
```

Listing 8.1: Kodeudsnit af setUp-metoden fra TestFacade-klassen

I *FacadeTest*-klasse testes metoderne `createEpisode()`, `approveProgram()` og `updateCredit()`. De to sidstnævnte er valgt, for at teste brugsmønstrene B11 *Opdater kreditering* og B13 *Godkend nyoprettet kreditering*, som der tidligere er stiftet bekendsskab med i rapporten. `approveProgram()`, som hører til brugsmønster B13 *Godkend nyoprettet kreditering*, testes, ved at godkende det første program i systemet, som er blevet importeret fra databasen, og bekræfte, at `isApproved()` på `program` er `True`. Se Listing 8.2.

```
48     @Test
49     public void approveProgram() {
50         facade.approveProgram(facade.getPrograms().get(0));
51         assertTrue(facade.getPrograms().get(0).isApproved());
52     }
```

Listing 8.2: Kodeudsnit af approveProgram-metoden fra TestFacade-klassen

Herefter testes metoden `updateCredit()`, Listing 8.3, som hører sammen med brugsmønster B11 *Opdater kreditering*. Her oprettes først et nyt objekt af typen `CreditedPerson`, Linje 58, og derefter en ny `Credit` for denne person, Linje 59. Derefter opdateres krediteringens rolle ved hjælp af `updateCredit()` i facade instansen, Linje 61. Til slut kontrolleres det at denne credits rolle er opdateret til den nye, Linje 62.

```
54     @Test
55     public void updateCredit() {
56         UUID creditedPersonUuid = UUID.randomUUID();
57         Program episode = facade.getProgramFromCreatedBy(
58             "TestEpisode", loginHandler.getUsers().get(0).getUuid()
59             ;
60         facade.createPerson(creditedPersonUuid, "TestPerson");
61         facade.createCredit(facade.getPersonFromUuid(
62             creditedPersonUuid), Credit.Function.VISUALARTIST,
63             episode);
64         Credit credit = episode.getCredits().get(0);
65         facade.updateCredit(credit, Credit.Function.CAST);
66         assertTrue(credit.getFunction() == Credit.Function.CAST);
67     }
```

Listing 8.3: Kodeudsnit af updateCredit()-metoden fra TestFacade-klassen

✓	✓ FacadeTest	508 ms
✓	✓ updateCredit	400 ms
✓	✓ createEpisode	56 ms
✓	✓ approveProgram	52 ms
✓	✓ LoginHandlerTest	2 ms
✓	✓ verifyCredentials	1 ms
✓	✓ createProducer	1 ms
✓	✓ ProgramTest	0 ms
✓	✓ deleteCredit	0 ms
✓	✓ addCredit	0 ms

Figur 8.1 Resultatet af unittestene

Desuden testes `createEpisode()` fra Facade-klassen, hvor der oprettes to ens instanser; en lokal instans og en i facden, og det kontrolleres at attributterne på instanserne er ens. I Program-klassen testes `addCredit()` hvor der tilføjes en credit til et program, og kontrolleres at credits på programmet ikke er null, og `deleteCredit()` hvor der fjernes en credit fra programmet og kontrolleres at det er null. I LoginHandler testes `createProducer()` hvor det kontrolleres at der tilføjes en producer i LoginHandleren og at denne ikke er null. Endeligt testes `verifyCredentials()`, hvor det kontrolleres, at dennes credentials returnerer False.

Unittestene køres ved brug af Maven og IntelliJ. Alle testene passer, og testene er vellykket, se Figur 8.1. Dette betyder at klasserne for sig selv fungerer som de skal.

8.2 UP Brugsmønstertest

Der er yderligere udført brugsmønstertest for brugsmønstrene B11 og B13. Testen af disse brugsmønstre foregår ved en gennemgang af brugsmønstrenes hovedhændelsesforløb punkt for punkt, som vil blive illustreret med skærmbilleder fra programmet.

8.2.1 Test af brugsmønster B11 Opdater kreditering

En detaljeret brugsmønsterbeskrivelse af brugsmønsteret B11 *Opdater kreditering* findes i Tabel 4.3 under Afsnit 4.2. Prækonditionerne for brugsmønsteret er, at programmet skal have en oprettet kreditering i systemet, og at aktøren har valgt et program. Dette ses på Figur 8.2, hvor transmissionen 'Top Gun' er markeret med grå, og transmissionens krediteringer vises i den nederste liste. Hovedhændelsesforløbets tre første punkter starter med, at aktøren vælger en kreditering fra programmet og trykker på opdater kreditering. Dette ses også på Figur 8.2, hvor den øverste kreditering er markeret med grå, og 'Update

KAPITEL 8. TEST

credit' knappen er highlightet med blå¹.

The screenshot shows a software interface for managing media content. On the left, there's a large red 'TV2' logo. The interface has several dropdown menus and buttons. At the top left, there's a 'Transmission' dropdown and a 'Choose TV-series' dropdown. To the right of these are buttons for 'Update TV-series', 'Search for program', and 'Update program'. In the center, there are two sections for TV series: 'Top Gun' (ID: 2a93e924-6638-465a-b5cd-cb8d16f00d69) and 'Flyvende Farmer' (ID: 3129a54d-2d5f-4fbc-a39a-9c478905ff11). Both sections include 'Created by: producer' links. Below the TV series are two credit entries: 'Mathias Engmark: Casting' and 'Christoffer Krath: Billed- og lydredigering'. To the right of these credits are buttons for 'Search for credit', 'Select function filter', 'Update credit' (which is highlighted in blue), and 'Update person'.

Figur 8.2 Test af punkt 1 til 3 i brugsmønster B11

Ved punkt 4, efter aktøren har trykket på 'Update credit' knappen, viser systemet den valgte kreditering for programmet, hvilket ses på Figur 8.3.

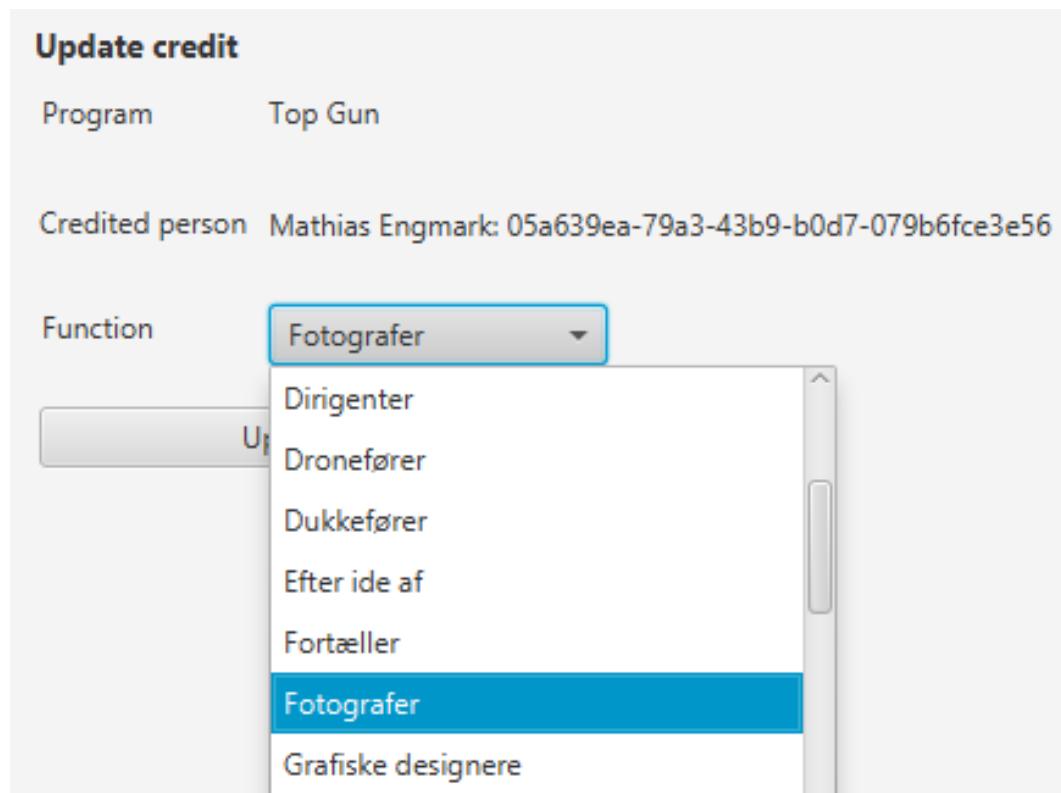
This screenshot shows a modal dialog titled 'Update credit'. Inside the dialog, there is a 'Program' dropdown menu with 'Top Gun' selected. Below it is a 'Credited person' field containing the text 'Mathias Engmark: 05a639ea-79a3-43b9-b0d7-079b6fce3e56'. Underneath that is a 'Function' dropdown menu with 'Casting' selected. At the bottom of the dialog is a large, prominent 'Update credit' button.

Figur 8.3 Test af punkt 4 i brugsmønster B11

Punkt 5 i hovedhændelsesforløbet går ud på, at aktøren opdaterer krediteringens felter med nye krediteringsoplysninger, hvilket ses på Figur 8.4, hvor krediteringens funktion

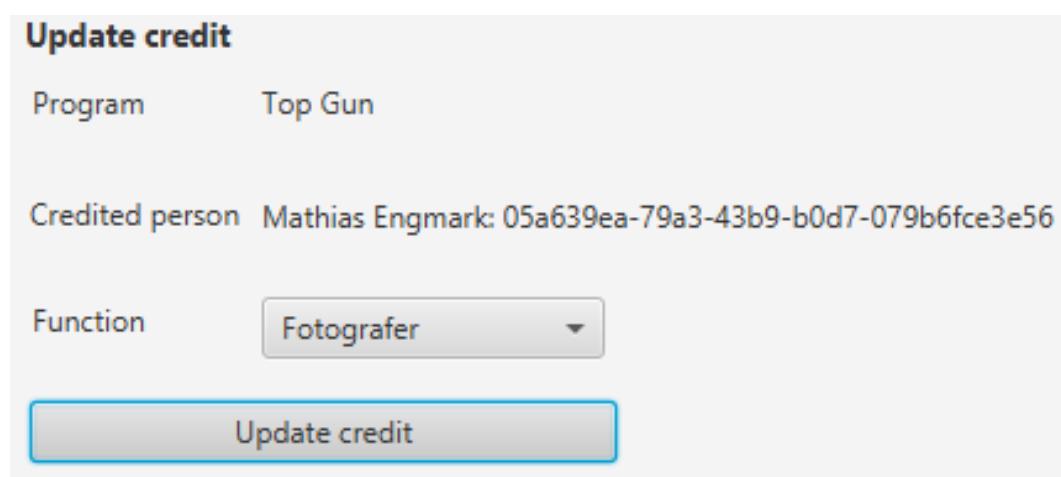
¹Som beskrevet ved implementeringen, Kapitel 7, er det kun muligt at bruge *Update Credit*-knappen, hvis den som er logget ind er af typen *producer* eller *systemadministrator*. I dette tilfælde hedder producenten faktisk *producer*, men de skal ikke forveksles med hinanden

ændres fra 'Casting' til 'Fotografer'.



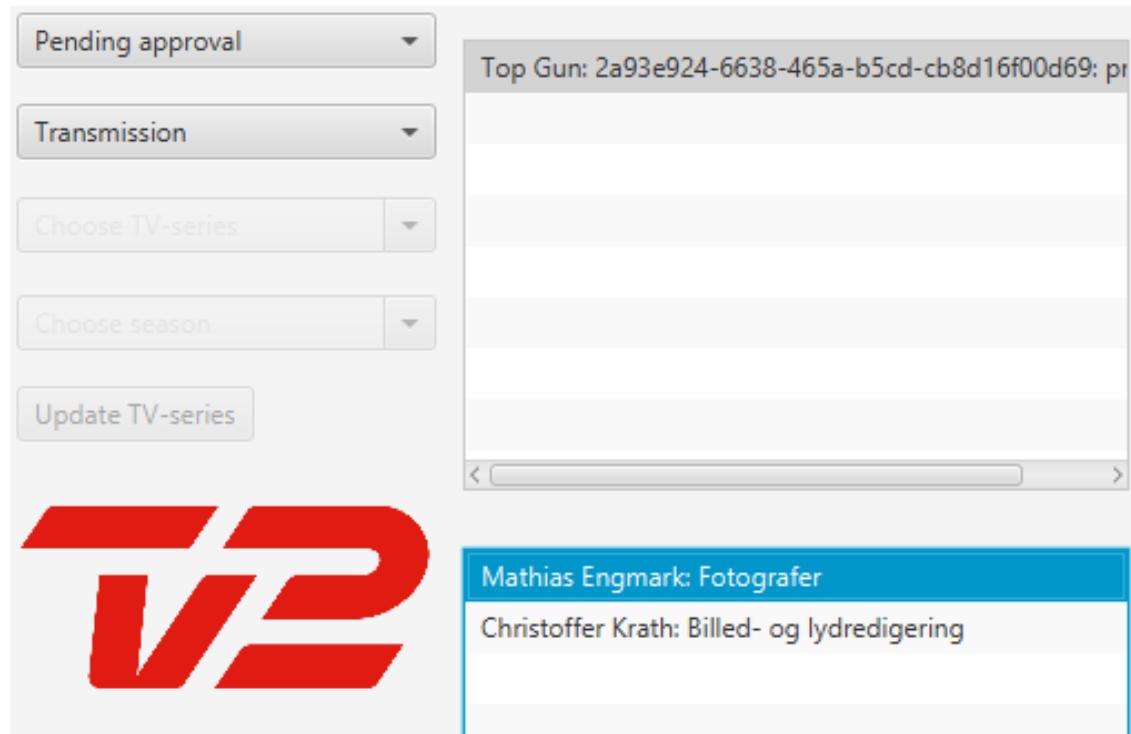
Figur 8.4 Test af punkt 5 i brugsmønster B11

Ved punkt 6 i hovedhændelsesforløbet trykker aktøren på 'Update credit' knappen fra *update*-tabben, som er vist highlightet med blå på Figur 8.5.



Figur 8.5 Test af punkt 6 i brugsmønster B11

Både punkt 7 og 8 samt brugsmønsterets to postkonditioner omhandler systemets reelle opdatering af krediteringen. Dertil omhandler de også, at programmet, der har fået ændret sine krediteringsoplysninger, er blevet opdateret til at afvente godkendelse af de gældende ændringer. Både krediteringen med de opdaterede oplysninger og programmet, der nu afventer godkendelse, kan ses på Figur 8.6. Her ses det, at transmissionen 'Top Gun' ligger under transmissioner, der afventer godkendelse, og at krediteringen har sine nye oplysninger i form af funktionen 'Fotografer'.



Figur 8.6 Test af punkt 7 og 8 i brugsmønster B11

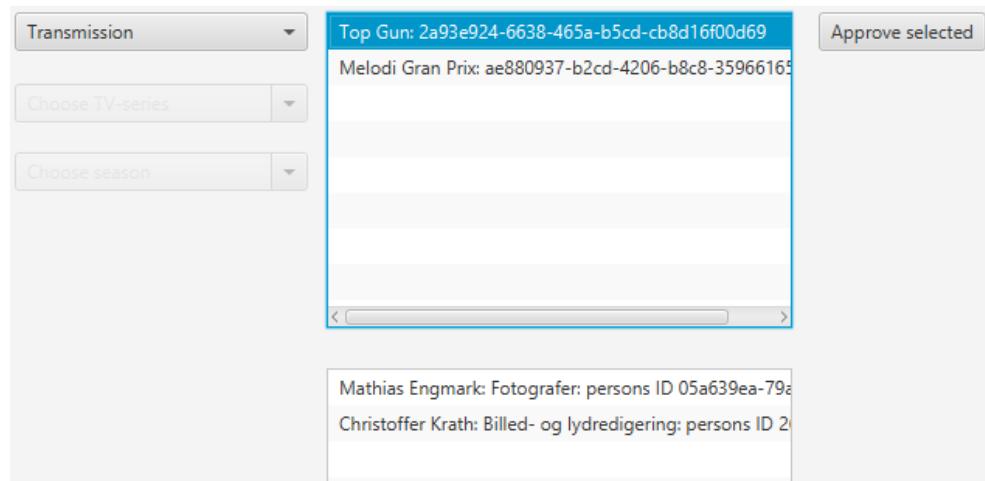
8.2.2 Test af brugsmønster B13 Godkend program

En detaljeret brugsmønsterbeskrivelse af brugsmønsteret B13 *Godkend program* findes i Tabel 4.4 under Afsnit 4.2. Brugsmønsterets prækonditioner og de to første punkter i hovedhændelsesforløbet beskriver, at systemet skal indeholde mindst et program, og at systemet viser en liste over programmerne, der afventer godkendelse. Dette ses i Figur 8.7, hvor listen indeholder transmissioner, der afventer godkendelse.



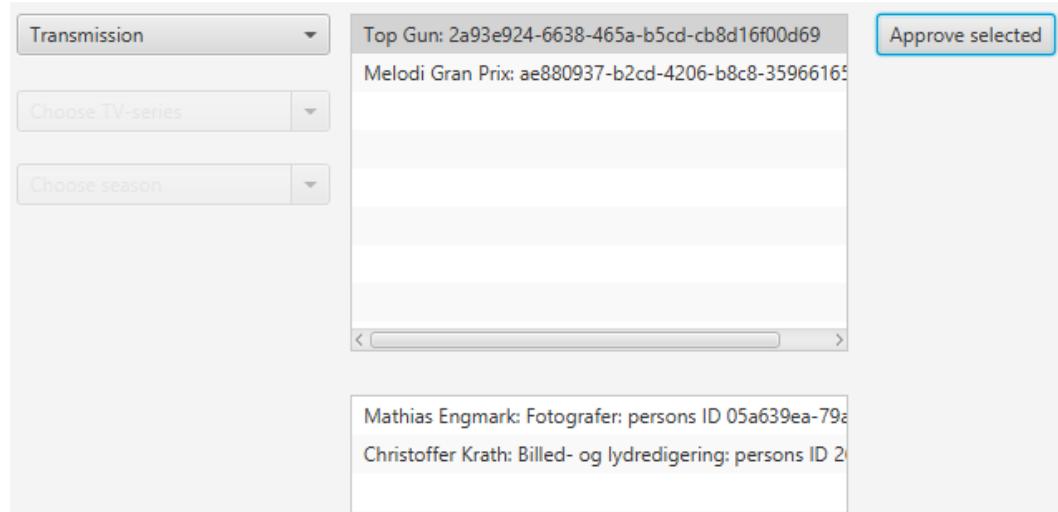
Figur 8.7 Test af punkt 1 og 2 i brugsmønster B13

Punkt 3 og 4 i hovedhændelsesforløbet går ud på, at aktøren vælger et af de afventede programmer, hvorefter systemet viser det valgte program og de tilhørende krediteringer. Dette ses på Figur 8.8, hvor transmissionen 'Top Gun', som også var testeksemplaret i Afsnit 8.2.1, highlightes med blå og får fremvist sine tilhørende krediteringer nedenunder i krediteringslisten. Her har punkt 4 også to underpunkter 4.1 og 4.2, som omhandler muligheden for at redigere programmets oplysninger før det godkendes. Denne funktionalitet er ikke blevet implementeret, qua projektet *kun* udvikles som en prototype.



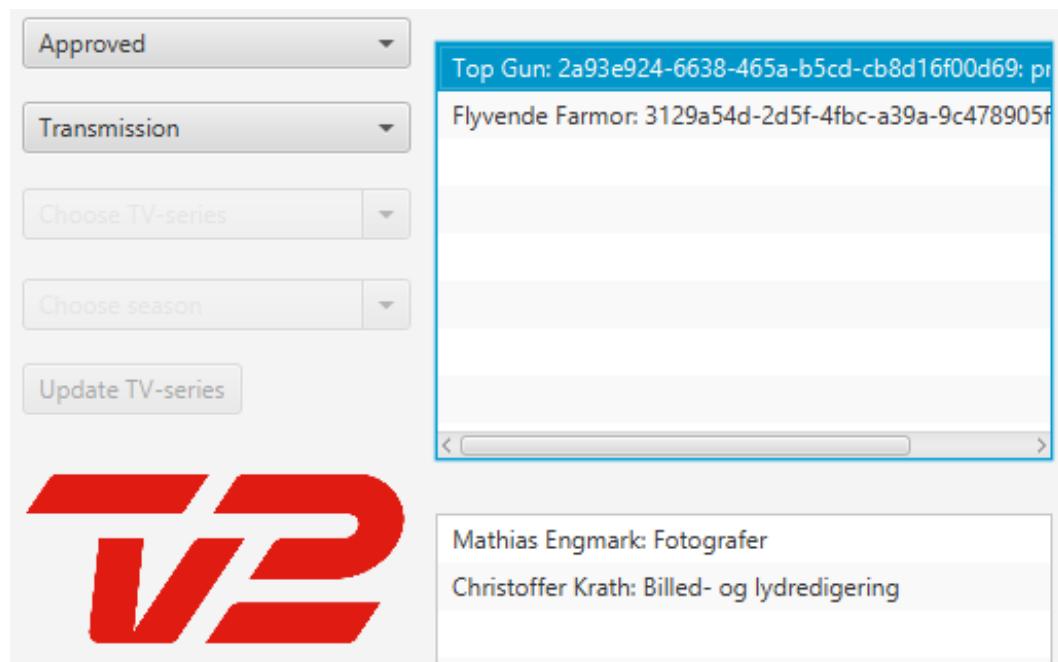
Figur 8.8 Test af punkt 3 og 4 i brugsmønster B13

I punkt 5 i hovedhændelsesforløbet godkender systemadministratoren det valgte program ved at trykke på 'Approve selected' knappen, hvilket ses på Figur 8.9.



Figur 8.9 Test af punkt 5 i brugsmønster B13

Punkt 6 og postkonditionen beskriver at programmet er blevet tilføjet til systemet, hvilket kan ses på Figur 8.10, hvor producenten, som har fået godkendt sit program, kan finde sit program under godkendte transmissioner. Her kan det også ses, at den foretagede ændring i den øverste kreditering er bibeholdt, da krediteringen har funktionen 'Fotografer' i stedet for 'Casting'.



Figur 8.10 Test af punkt 6 i brugsmønster B13

Brugsmønsteret har også et alternativt hændelsesforløb ved punkt 5, hvor systemadministratoren kan vælge at afvise programmet, hvilket ville resulterer i at programmet ikke godkendes eller tilføjes. Dette alternative hændelsesforløb er dog heller ikke blevet implementeret af samme årsag.

8.3 Brugertest

For at få en forståelse af, om programmet opfylder dets formål opstilles scenarier som er meningen skal dække programmets primære brugsmønstre². Da gruppen ikke kan teste med kunden selv (TV2) har gruppen testet med personer udenfor gruppen og studiet, som ikke har sat sig ind i projektcasen. Gruppen observerer om testpersonen gør det opstillet scenario efter hensigten, mens testpersonen forklarer hvad han/hun gør. Efterfølgende gives feedback fra testpersonen i forhold til oplevelsen. De scenarier der stilles er som følger:

- Start som gæst
- Find "Chicago Fire", Sæson 2 Episode 1, "Everything Burns" i systemet
- Log ud
- Log ind som producer (Brugernavn: "producer", Adgangskode: "producer")
- Opret et program, giv det et navn og udfyld resten af informationen som det giver intuitivt mening for dig
- Opret en person og en kreditering der hører til det program
- Log ind som systemadministrator (Brugernavn: "admin", adgangskode: "admin")
- Godkend krediteringerne på programmet
- Opdater det oprettet program og giv det et nyt navn som Systemadministrator

Ud fra ovenstående fremgangsmåde er der anskaffet feedback fra to testpersoner. Se Tabel 8.1 for feedback fra testperson 1, og Bilag E Tabel 11 for feedback fra testperson 2.

Spørgsmål	Svar
Hvordan var din oplevelse med at logge ind som hhv. gæst og producer/system administrator?	Nem og brugervenlig.

²I henhold til gruppens Must-haves fra Tabel 4.1 Kapitel 4

Hvordan var din oplevelse med at søge og vise programmer og deres krediteringer som gæst? Er informationen præsenteret ordentligt? Hvad kunne gøres for at forbedre dette?	Intuitivt at søge i, og informationen hører til de rigtige kategorier ved eftersyn.
Hvordan var din oplevelse med at oprette et program og krediteringer til programmet? Er brugergrænsefladen forståelig og tilfredsstillende intuitiv? Er det overskueligt og pænt? Har du oplevet nogle fejl eller mangler? Hvordan er din forståelse af hvad programtyperne ville være i den virkelig verden? Giv gerne et eksempel på hvad de hver kunne være (transmission, episode og tv-serie)	UI er intuitivt at navigere i, der er orden i brugergrænsefladen ved første blik. UI skal læses før man benytter den, ellers kan der komme meget kortvarig forvirring. Enkelte knapper ovenpå hinanden, men få af dem. Transmission ville svare til TV2's nyhedsudsendelser som eksempel. Episoder og TV-serie kan f.eks. en være en dramaserie, hvor der under denne serie er knyttet episoder.
Hvordan var din oplevelse med at opdatere programmer? Oplevede du nogle fejl eller mangler? Var det en intuitiv oplevelse?	Øjeblikkelig opdatering og igen er det overskueligt visuelt.
Hvordan var din oplevelse med at godkende programmer som system administrator? Hvordan er din forståelse af hvad der sker, når man godkender et program?	Med det samme forstås, at programmet bliver "aktuelt", og er nu synligt for seerne.
Har du andre bemærkninger?	-

Tabel 8.1 Feedback fra testperson nr. 1

I grove træk, viser denne feedback og ovennævnte feedback Bilag E, at der generelt er en tilfredsstillende grad af brugervenlighed og intuitivitet i programmet, hvilket også var en del af de supplerende krav i tabel 4.2 i Kapitel 4. Med det sagt, betyder det ikke, at programmet er fejlfrit og færdigt, men derimod at resultatet er tilfredsstillende i forhold til kravene og brugsmønstrene. Feedbacken ville give anledning til en 3. iteration, hvor brugsmønstrene og kravene ville blive revurderet og implementeringen eventuelt opdateret.

Der blev blandt andet testet i forhold til brugsmønstrene B11 *Opdater kreditering* og B13 *Godkend nyoprettet kreditering*, hvor det, ud fra observationerne, kan vurderes, at der fra ingen af testpersonernes side var problemer med at udføre disse brugsmønstre.

KAPITEL 9

Diskussion

Målet med projektet er, at udvikle et Credits Management System, som kan afskaffe TV2's krediteringstid på programmer, samtidig med, at minimere arbejdsprocessen udført af TV2, beskrevet ved Kapitel 1. Systemet havde til hensigt, at blive udviklet som en prototype, som blandt andet indeholder muligheden for persistering af data - og i det aspekt, betragtes projektet som en succes.

Gruppen har blandt andet opstillet en række krav, Kapitel 4 Figur 4.1, som yderligere definerer hvilke egenskaber systemet skal besidde, hvilket blev prioriteret gennem MoSCoW-metoden. På baggrund af prioriteringen, er der udarbejdet en række unittests, brugsmönstertest og brugertest, Kapitel 8, som skulle validere hvorvidt de enkelte krav er opfyldt, og på baggrund deraf, vurderes det at projektets *must-have* krav, fra Figur 4.1, er tilstrækkelig implementeret. Dette betyder ikke, at systemets implementering er perfekt. Tværtimod er systemet udarbejdet som en prototype, hvilket også betyder, at systemets funktionalitet kan forbedres. Det kommer eksempelvis til udtryk ved Kapitel 8.2, hvor blandt andet brugsmønster B13 punkt 4.1 og 4.2 ikke er implementeret. Selvom der mangler reel implementering i henhold til brugsmönstrene, vurderes det ikke at være en mangel for at projektets *must-have*-krav fungerer - ej heller for projektets formål; at minimere arbejdsprocessen udført af TV2 og at frigøre krediteringstiden efter et udsendt program.

Det at arbejdsprocessen bliver minimeret kommer til udtryk gennem designet af, hvordan et program og dertilhørende krediteringer bliver oprettet i systemet. Som beskrevet ved indledningen, Kapitel 1, oprettes omtalte, på nuværende tidspunkt, ved at en producer sender et allerede oprettet dokument til TV2, som efterfølgende selv indskriver dokumentet. Det udviklede system fjerner helt eller delvist den arbejdsproces, ved at muliggøre at en producer direkte kan indskrive omtalte i systemet, hvor TV2 kun skal godkende

programmet og dets krediteringer efterfølgende. At krediteringstiden bliver frigjort kan diskuteres, for på nuværende tidspunkt belyser projektet ikke, hvordan TV2 formelt fjerner krediteringerne fra fjernsynet, men derimod blyses det, hvordan et muligt system kan designes og udvikles. Det ses ikke nødvendigvis som en fejl, hvis der henholdes til projektet egentlige problemformulering; *"Hvordan kan man opstille et Credits Management System, som gør det muligt for TV2 at afskaffe krediteringstiden på programmer?"*.

Gruppen har endvidere udarbejdet en brugertest, Kapitel 8.3, som havde til hensigt at vise, hvorvidt systemet lever op til dets formål. Testen, som blev udarbejdet ud fra nogle scenarier, som skal efterligne den egentlige brug af systemet, viste en tilstrækkelig forståelse for, hvordan systemet fungerer.

Selvom brugertesten kun blev udført på to testpersoner, viste der sig en tendens; at brugervenligheden var til stede. Brugervenlighed er også en af de supplerende krav beskrevet i Kapitel 4 ved Figur 4.2. Selvom brugervenligheden virkede tilstrækkelig overskuelig, oplevede testpersonerne også en del fejl og mangler. Blandt andet pointerede én af testpersonerne at søgefeltet ikke var tilstrækkeligt stort, mens en anden savnede feedback fra systemet, når handlinger, som eksempelvis *create person*, blev udført. Testen i sin helhed viste altså, at projektets system er brugbart, dog med mulighed for forbedring, hvilket også giver mening, eftersom systemet er udviklet som en prototype. Feedbacken fra testen, kunne have været brugt til netop forbedring af brugeroplevelsen, hvis gruppens opgave var at udvikle et fuldt funktionsdygtigt system.

Udviklingen af systemet er dels sket på baggrund af projektets case, som kort er beskrevet ved indledningen, Kapitel 1, samt på baggrund af krav og analysen ved Kapitel 4 & 5. Gruppen kunne opnå yderligere og eventuelt bedre resultater, hvis det udviklede system blev testet direkte af TV2, qua en mere brugerdefineret feedback fra den egentlige kunde, hvilket vil blive yderligere redegjort ved Kapitel 11.

KAPITEL 10

Konklusion

Systemet er lavet ved brug af objektorienteret programmering, har en brugervenlig brugergrænseflade, og er forbundet til en relationel database lavet i PostgreSQL. Systemet er endvidere designet med tre forskellige roller in mente, gæster, producere og systemadministrator, der repræsenterer henholdsvis TV-seere, filmselskaberne og TV2. Disse har hver deres funktionalitet og formål ved brug af systemet, som ønsket af TV2.

De opstillede krav, analysen af dem samt designet af systemet, har dannet grundlag for udviklingen af et Credits Management System, hvor fokuspunktet for udviklingen har været i henhold til afskaffelse af krediteringstiden på programmer. På baggrund af de udarbejdet test, som tager udgangspunkt i systemets krav, kan det konkluderes, at systemet lever op til muligheden for afskaffelse af krediteringstiden, men i takt med, at produktet kun har været udviklet som en prototype, er der mulighed for forbedring af systemets funktionalitet, hvilket brugsmønster- og brugertesten også synliggøre. Ydermere kan det, i henhold til systemets implementering, konkluderes, at der er mulighed for at fjerne den dobbelte arbejdsproces, som der på nuværende tidspunkt finder sted ved indskrivning af krediteringer hos TV2.

Det kan endvidere konkluderes, at det vurderes sandsynligt, at en test, udført direkte på kunden, TV2, kan føre til andre og eventuelt bedre resultater i henhold til opstillingen af et Credits Management System via en mere brugerdefineret feedback. Men resultatet af det udviklede system, kan stadig vurderes som et muligt løsningsforslag for eventuel fremtidig udvikling af systemet.

KAPITEL 11

Perspektivering

Credits Management Systemet, som har til formål at samle krediteringerne fra TV2 ét sted, kunne ses anvendt hos andre TV-selskaber, eller måske endda som et samarbejde mellem TV-selskaberne, mens den udviklede prototype, som projektet dækker over, kunne ses som grundlag for videreudviklingen af et fremtidigt system. Hvis projektgruppen stod for en eventuel videreudvikling af systemet, ville hyppigere møder med kunden være yderst relevant, for at brugerdefinere systemet mere, og derved holde kunden for øje under udviklingen. Det vil samtidig give muligheden for at teste direkte på kunden, som kort blev nævnt ved Kapitel 9, og derved få en feedback fra dem, som skal bruge systemet dagligt.

Ved en eventuel videreudvikling af det udviklede produkt, kunne implementering af de krav fra kapitel 4, som ikke er blevet implementeret, holdes for øje. Eventuelt i henhold til feedback direkte fra TV2. Det kunne ydermere være af interesse at videreudvikle på programmet ved at bruge teknologier, som ligger uden for projektets faglighed, defineret ved Kapitel 2.3, til bedre håndtering af login, datahåndtering eller at gøre brug af TV2's SSO system og client-server modellen. Ligesom en implementering, som muliggør at brugeren nemmere kan finde udsendelser som han/hun lige har set hos TV2, fx via integrering med deres platform TV2 Play eller en QR-kode i slutningen af udsendelsen, der ville tage brugeren til en webplatform, hvor Credits Management system bliver hostet.

Ydermere kunne det være af interesse at lade programmet håndtere mere information om dets programmer. Eksempelvis ved en yderligere detaljegrad af credits, som tager højde for skuespillere og deres roller i film/serier, samt eventuelt en *skuespillerprofil*, som indeholder information om en given skuespille. Information relevant og kun synlig for headhunters, som kunne gøre brug af systemet, for at finde en skuespiller eller andre medvirkende der passede ind i deres produktion.

KAPITEL 12

Procesevaluering

Gruppens tanker om projektet samt refleksion over arbejdsprocesser og -form vil blive belyst. Endvidere vil der blive evalueret på gruppens egen indsats samt projektets omfang og gruppens tilgang til opgaver stillet af projektet.

12.1 Samarbejdet i gruppen

Som udgangspunkt har samarbejdet i gruppen været særligt eminent, og der har ikke været nogle diskussioner eller uenigheder af særlig betydning. Alle gruppemedlemmer har tildels haft samme forventningsniveau til arbejdsbyrde, hvilket har vist sig særligt nyttigt ved projektets planlæggelse, samt ved fastlæggelse af gruppen arbejdstider. Langt hen ad vejen har gruppen også arbejdet fælles med mange af de opgaver, som var påkrævet af projektet såsom fælles diskussion af design- & analysemæssige opgaver, men også fælles programmering, hvor der har været anvendt par- & teamprogrammering for at give alle i gruppen en god grundforståelse af programmet. I gruppens teamprofil fra Bilag K Afsnit 9.3, konkluderende gruppen, på baggrund af Belbins teamroller, at gruppemedlemmer kunne *"have tilbøjelighed til at være uflexible og stædige"*, baseret på en overvægt i organisator rollen. Dette blev mitigeret ved at vi var opmærksomme på at gå på kompromis og på den måde afgøre diskussioner i fællesskab.

12.2 Samarbejde med vejleder

Samarbejdet med vejleder har som udgangspunkt været godt, men som gruppe kunne vi have ydet fordel af, bedre at have udnyttet vejlederen i forhold til mere feedback ved at sende materiale oftere. Det har været en fordel for vores gruppe, at vejledermøderne, bortset fra i starten, altid har været samme tidspunkt på samme dag - det var ikke alle grupper der havde den luksus. Det skabte en vis forudsigtelighed, som støttede projektets

planlægning.

Til trods for den gode vejleder, er der enighed i gruppen om, at det har været problematisk, at der flere gange har været inkonsistens i information modtaget fra henholdsvis vejleder og undervisere, og det har ført til unødvendig forvirring.

12.3 Læringsprocessen

Gruppen har med semesterprojektet fulgt undervisningen i Grundlæggende Software Engineering(GSE), Data Management samt Videregående Objektorienteret programmering, og har udarbejdet projektet ud fra lektionerne på ugentlig basis. Projektet var struktureret meget ud fra netop GSE undervisning, hvilket har givet mange læringsmæssige fordele, qua at gruppen har kunne bruge den teoretiske viden i praksis i projektet. Derudover gav dette også en struktur for, hvordan projektet skulle udarbejdes, da undervisningen kunne sætte rammer for, hvad der skulle arbejdes med uge for uge.

Som nævnt ved Kapitel 12.1, har gruppen langt hen ad vejen arbejdet fælles om opgaverne, hvilket har hjulpet indlæringen; at man kan sparre med de andre gruppemedlemmer og diskutere, hvorfor én løsning kunne være bedre end en anden i en given opgave. Under programmering og implementering af det udviklede system, fik alle også på skift mulighed for at programmere med input fra de andre gruppemedlemmer. Der har i det hele taget været en god forståelse i gruppen for, at ikke alle resultater vil være korrekte i første omgang, og at hele semesterprojektet også skal ses som en læringsproces såvel som andet undervisning skal.

12.4 Projektarbejdsform

Gennem projektet har gruppen fulgt en meget fleksibel arbejdsform. Typisk mødtes gruppen et par gange hen over starten af projektet, hvor vi midtvejs, hvor implementeringen begyndte, mødtes flere gange om ugen, hvor opgaverne blev fordelt ligeligt i grupper eller lavet fælles. Dette var specielt synligt ved implementering, hvor vi benyttede os meget af par- og teamprogrammering. Efter 1. iteration af projektet¹, delte vi os op i to- eller tremandsgrupper, hvor det der skulle laves, baseret på gruppens productbacklog, blev fordele mellem grupperne. I løbet af implementering har vi skiftes til at skrive, samt være den som fortalte hvad der skulle skrives. Parprogrammering fungerede rigtig godt for os, da alle fik muligheden for at få fingrene i koden, samtidig med at gøre arbej-

¹1. iteration dækker ind til midtveisseminalaret, hvor vi skulle aflevere første del af rapporten og koden

det lettere, ved at være flere om opgaverne. Gruppen har også gjort brug af pararbejde i analyse- og designfasen for at fordele opgaverne, og skabe en fagligdiskussion om arbejdet.

Gruppen har igennem projektet benyttet sig af en logbog, hvori der for hver arbejdsdag blev skrevet et kort oprids af, hvad der blev lavet den pågældende dag. Gruppen kunne med fordel have brugt logbogen mere effektivt, eftersom der i sidste ende kun var én mand, som skrev i den. Gruppen har også indset, at det ville have været optimalt at notere refleksioner samt de erfaringer vi har opnået gennem projektet.

Gruppen har desuden arbejdet med Jira, og på denne måde tracket vores opgaver. Her har vi oprettet issues om analyse og design af brugsmønstre, implementering af brugsmønstre og rapportafsnit. Her har vi arbejdet med 6 forskellige kolonner til vores issues: *Backlog*, *in-progress*, *bottle neck/haster*, *gået i stå*, *review* og *done*. Meningen er, at optimalt skal en issue gå fra backlog, til in-progres, til review til done, men de andre kolonner gøres brug af, når en issue er for svær for den assignede person, eller hvis en issue skal implementeres før en anden kan komme videre. Dette har været meget brugbart, specielt i forbindelse med Scrum. Her har product owneren stået for at styre product- og sprintbackloggen i Jira. Gruppen har kun arbejdet i to sprints, hvilket er længere end 30 dage, som ellers er maxanbefalingen, men dette forsvarer med, at gruppen ikke har arbejdet fuldtid på projektet. Gruppen har lavet sprint planning, hvor issues fra product backloggen er blevet inddelt i sprintet, og har også dagligt talt om og planlagt issues fra sprint backloggen i fællesskab i noget lignende et daily scrum, selvom det ikke blev gjort ligeså formelt.

12.5 Hvad kunne være gjort bedre?

Som det blev nævnt ved Kapitel 12.4, gjorde gruppen brug af en logbogen, som dokumentere processer undervejs. Efter en nærmere refleksion på brugen, vurderes det af gruppen, at der med fordel kunne have gjort yderligere brug af logbogen, qua vi i afsluttende faser, ikke behøvede at diskutere, hvad der var sket, men istedet kunne forholde os til noget som allerede var dokumenteret.

I implementationen af 1. iteration var det ikke meningen at projektet skulle indeholde en PostgreSQL database, men i stedet en lokal database i andet format. Vi valgte at gå med en løsning som gemte og læste relevant data fra .txt format ud fra en prædefineret standard opsat af gruppen. Problemet med denne metode var, at risikoen for at læse forkert data, eller endda slet ikke læse noget, var ret stor eftersom små ændringer, som ikke fulgte de prædefineret standarder ville gøre filen ulæselig. Grunden til valget af .txt

format lå på, at vi selv gerne ville kunne læse filerne når programmet ikke kørte, men efter en refleksion over valget kunne vi måske have anvendt Javas "Serialization"-funktion, og blot gemme direkte som objekter. Dette ville sikre ensartet opbevaring af data, samt gøre det lettere for programmet at læse. Eneste minus ved denne tilgang er, at vi som brugere ikke ville kunne tyde filerne udenfor programmet, men isoleret set ville det ikke gøre noget for programmet.

Der er også enighed i gruppen, at det ville have været godt, hvis man tidligere i processen havde kigget på udarbejdelse af rapporten, og fjerne noget af arbejdsbyrden fra den afsluttende del af projektet. Samtidig vil have været mere optimalt at bruge vejleder mere, som nævnt ved Kapitel 12.2, ved at gruppen fik skrevet dele af rapporten tidligere i projektet, fremfor de sidste to uger af projektperioden, og sendt afsnit løbende med henblik på feedback.

Bibliografi

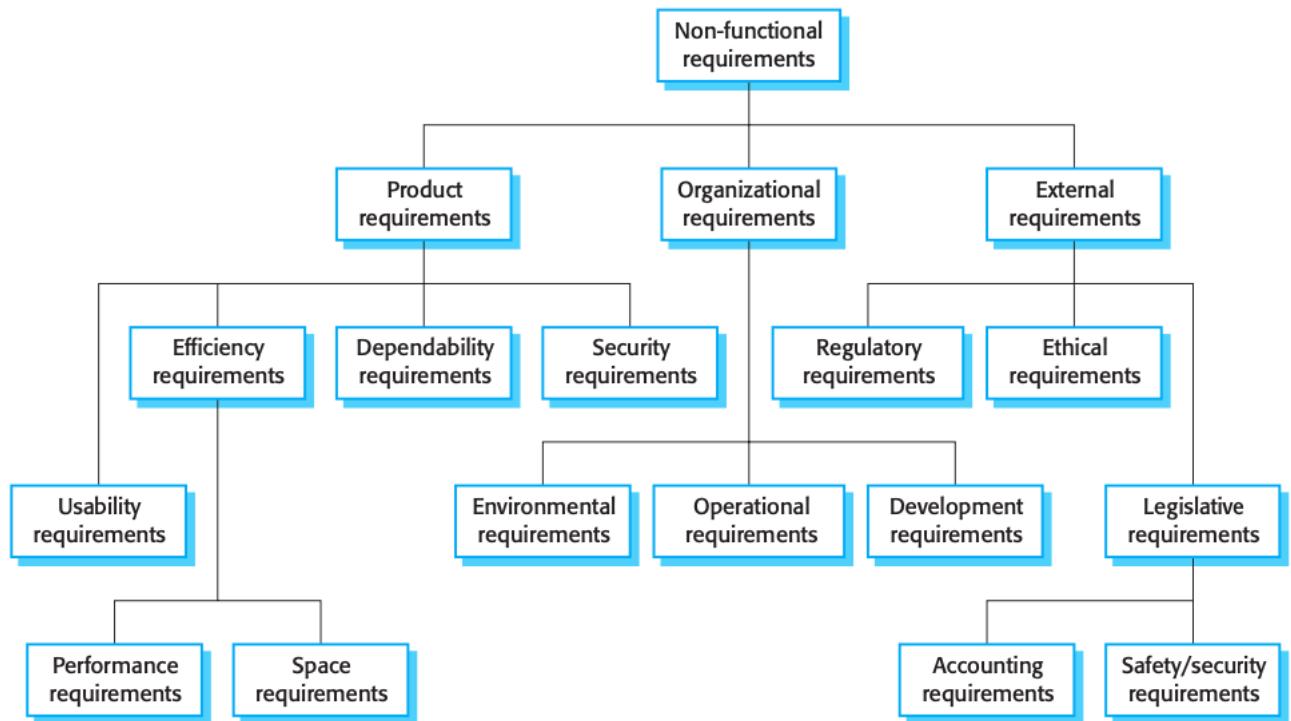
- [1] Kaya Ismail. *Kanban vs. Gantt Charts: Which Is Best for Your Team?* URL: <https://www.cmswire.com/information-management/kanban-vs-gantt-charts-which-is-best-for-your-team/>. (accessed: 29.05-2021).
- [2] Lone Borgersen. *Analysis Structure*. URL: https://282.kultura.nordu.net/media/t/0_3hift1hg/395105. (accessed: 24.05-2021).
- [3] Lone Borgersen. *Analysis Behavior*. URL: https://drive.google.com/file/d/1NF2p27Tslnb_EX8VvmGX2Zov3yqz5TtV/view. (accessed: 24.05-2021).
- [4] Izenda Editorial Staff. *5 Benefits of a 3-Tier Architecture*. URL: <https://www.izenda.com/5-benefits-3-tier-architecture>. (accessed: 19.05-2021).
- [5] IBM Cloud Education. *Three-Tier Architecture*. URL: <https://www.ibm.com/cloud/learn/three-tier-architecture>. (accessed: 26.05-2021).
- [6] Guido Dechamps Domenique Tilleuil. *The Importance of the Dependency Inversion Principle*. URL: <https://www.triple.d.io/07/05/2019/dependency-inversion-principle>. (accessed: 27.05-2021).
- [7] Ian Sommerville. „Software Engineering 10th Edition“. I: Pearson Education Limited, 2016. Kap. 7.1.4. ISBN: 1-292-09613-6.
- [8] W3School. *SQL Injection*. URL: https://www.w3schools.com/sql/sql_injection.asp. (accessed: 20.05-2021).
- [9] postgresqltutorial.com. *PostgreSQL DROP DATABASE*. URL: <https://www.postgresqltutorial.com/postgresql-drop-database/>. (accessed: 27.05-2021).
- [10] techopedia.com. *What is Atomicity*. URL: <https://www.techopedia.com/definition/24729/atomicity>. (accessed: 27.05-2021).
- [11] Ian Sommerville. *Software Engineering 10th edition*. Pearson Education Limited, 2016, s. 810. (accessed: 29.5.2021).

Appendices

Bilag

A	Typer af supplerende krav	61
B	Detaljeret brugsmønstre	61
C	Brugsmønsterrealiseringer	69
D	Fuldt SQL-Script	76
E	Brugertest: Feedback	82
F	Brugervejledning	83
G	Logbog	101
H	Designklassediagram	101
I	Oversigt over kildekode	101
J	Udfyldt rapportKontrolskema	101
K	Inceptionsdokument	107

A Typer af supplerende krav



Figur 1 Typer af ikke-funktionelle krav - Taget fra Software Engineering 10th edition Kapitel 4.1.2
Figur 4.3[11]

B Detaljeret brugsmønstre

Brugsmønster: Søge efter kreditering	
ID: B01	
Primære aktører: Bruger, Gæst	
Sekundære aktører: Mindst ét program skal være tilstede og godkendt i henhold til B13	
Kort beskrivelse: En bruger eller gæst - omtales aktør - skal søge efter et program eller en kreditering (eksempelvis skuespiller, producer eller fotograf), få en liste af søgeresultater der matcher kriterierne, og vælge det ønskede resultat.	
Prækonditioner (Pre-conditions):	<ul style="list-style-type: none"> • Ingen

Hovedhændelsesforløb (main flow):

1. Brugsmønsteret starter når aktøren vil søge på et program eller krediteret person
2. Aktøren kan vælge ønsket søgekategori
 - 2.1. Følgende søgekategori består af:
 - 2.1.1. Serie, film, program
 - 2.1.2. Krediteringsrolle
3. Aktøren søger på søgningskriteriet
4. Systemet laver et kald til databasen med det pågældende søgekriterie
5. Systemet modtager et svar fra databasen
6. Systemet viser relevante oplysninger efter det ønskede kriterier til aktøren

Postkonditioner (Post conditions):

- Søgekriteriet gemmes til brug i databehandling
- Aktøren får fremvist relevante oplysninger baseret på de givne søgekriterier.

Alternative hændelsesforløb (Alternative flow):

- Aktøren kan afslutte brugsmønsteret fra punkt 1-3
- Kreditering med søgekriteriet findes ikke
- Hvis forbindelsen til databasen eller systemet afbrydes før resultatet modtages, får aktøren besked

Tabel 1 Detaljeret brugsmønster - søge efter kreditering

Brugsmønster: Logge ind	
ID:	B04
Primære aktører:	Gæst
Sekundære aktører:	Ingen
Kort beskrivelse: En gæst skal med korrekte legitimationsoplysninger kunne logge ind på systemet og tilgå deres personlige konto. <i>Det antages at systemet ser en hver, som ikke er logget ind, som en gæst</i>	
Prækonditioner (Pre-conditions):	
<ul style="list-style-type: none"> • Gæst har åbnet applikationen 	

Hovedhændelsesforløb (main flow):

1. Brugsmønsteret starter, når gæst gerne vil logge ind på systemet
2. Systemet viser et loginvindue
3. Gæst indtaster brugernavn og adgangskode
 - 3.1. , Hvis en gæst ikke har en konto: B05 Oprette en bruger «extends»
4. Gæst trykker på “log ind”
5. Systemet ændrer viewet baseret på gæsts personlige konto
6. Gæst er logget ind på sin personlige konto

Postkonditioner (Post conditions):

- Gæst er logget ind på sin personlige konto
- Systemet viser det korrekte systemview

Alternative hændelsesforløb (Alternative flow):

- Ved punkt 2-3: Gæst kan vælge at afslutte brugsmønsteret
- Ved punkt 5: Hvis brugernavn ikke eksisterer i systemet eller hvis brugernavn og adgangskode ikke stemmer overens, vil gæst ikke blive logget ind
 - Systemet viser beskeden “De indtastede loginoplysninger er forkerte”
- Ved punkt 3.1: Hvis gæst ikke har en eksisterende bruger i systemet vil gæst kunne oprette en bruger

Tabel 2 Detaljeret brugsmønster - Logge ind

Brugsmønster: Logge ud
ID: B06
Primære aktører: Bruger, Producer & Systemadministrator. (Omtales aktør fremover))
Sekundære aktører: Ingen
Kort beskrivelse: En aktør, som er logget ind, skal kunne logge ud igen
Prækonditioner (Pre-conditions):
<ul style="list-style-type: none"> • Aktøren skal være logget ind

Hovedhændelsesforløb (main flow):
1. Brugsmønstret starter, når aktøren gerne vil logge ud.
2. Aktøren trykker på knappen log ud
3. Systemet viser et pop-up-vindue med bekræftelse om at logge ud
4. Aktøren bekræfter valget
5. Systemet logger aktøren ud
Postkonditioner (Post conditions):
• Aktøren er logget ud fra sin konto, og aktørens systemview er ændret til gæst
Alternative hændelsesforløb (Alternative flow):
• Aktøren kan afbryde brugsmønstret fra punkt 2-4
• Ved punkt 4: Hvis brugeren trykker "nej" til bekræftelsen stoppes brugsmønsteret

Tabel 3 Detaljeret brugsmønster - Logge ud

Brugsmønster: Oprette program med kreditering
ID: B10
Primære aktører: Producer, systemadministrator
Sekundære aktører: EPG-systemet
Kort beskrivelse: En producer skal oprette et program i systemet. Når programmet oprettes, skal der tilføjes relevant programoplysninger, og tilføjes relevante krediteringersoplysninger. Derefter sendes en forespørgsel til systemadministrator, som skal godkende krediteringen.
Prækonditioner (Pre-conditions):
• Producer skal være logget ind

Hovedhændelsesforløb (main flow):

1. Brugsmønstret starter, når producer skal oprette en kreditering
2. Produceren trykket på knappen “Opret ny kreditering”
3. Systemet kræver producerens log-in igen (include B02 (Login))
4. Systemet fører produceren til et nyt vindue, hvor krediteringer kan oprettes
5. Produceren udfylder de relevante felter og krediteringer
6. Produceren trykket på knappen “Opret kreditering”
7. Systemet sender de oprettede krediteringer til databasen
8. Systemet viser producerens oprettet kreditering
 - 8.1. Vises med en status som “afventer”
9. Systemet gemmer krediteringen og sender en forespørgsel til systemadministrator
10. Include B13 (Godkende program)

Postkonditioner (Post conditions):

- Den oprettede kreditering skal godkendes af systemadministratoren

Alternative hændelsesforløb (Alternative flow):

- Systemadministratoren kan afslutte brugsmønsteret overalt fra punkt 2-6
- Hvis producerens login (punkt 3) ikke er korrekt, afbrydes brugsmønster
- Hvis forbindelsen til systemet afbrydes under oprettelsen, får produceren besked og brugsmønsteret afbrydes
- Hvis forbindelsen til databasen afbrydes, får produceren besked, og brugsmønsteret afbrydes
- Produceren vil gerne stoppe oprettelsen midlertidigt (Punkt 5).
 - Produceren trykker på knappen “gem” under processen i punkt 5
 - Ved tryk på knappen “gem” gemmer systemet indskrevet data fra felterne

Tabel 4 Detaljeret brugsmønster - oprette program med kreditering

Brugsmønster: Eksportering af kreditering
ID: B12
Primære aktører: Systemadministrator
Sekundære aktører: Ingen

<p>Kort beskrivelse: Systemadministrator skal kunne eksportere krediteringer til manuel brug af dataanalyse</p>
<p>Prækonditioner (Pre-conditions):</p> <ul style="list-style-type: none">• Systemadministratoren skal være logget ind• Der skal eksistere krediteringer i systemet
<p>Hovedhændelsesforløb (main flow):</p> <ol style="list-style-type: none">1. Brugsmønsteret starter når systemadministratoren skal eksportere en eller flere krediteringer2. Systemadministratoren indtaster ønskede søgekriterie.3. Systemadministratoren søger efter ønskede krediteringer4. Systemet sender søgningen til databasen5. Systemet modtager data fra databasen6. Systemet viser det til systemadministratoren7. Systemadministrator markerer viste kreditering8. Systemadministrator eksporterer markerede krediteringer<ul style="list-style-type: none">8.1. Systemadministrator vælger ønsket filformat og filplacering9. Systemet eksporterer viste markering i ønsket filformat til ønsket filplacering
<p>Postkonditioner (Post conditions):</p> <ul style="list-style-type: none">• Systemet opretter og gemmer systemadministratorens handlinger i systemet.
<p>Alternative hændelsesforløb (Alternative flow):</p> <ul style="list-style-type: none">• Systemadministratoren kan afslutte brugsmønsteret fra punkt 1-8• Kreditering med søgekriterie findes ikke.• Hvis forbindelsen til databasen eller systemet afbrydes før resultatet modtages, får aktøren besked• Hvis filformat og placering ikke bliver angivet, sættes dette til en standard værdi• Hvis systemadministrator ikke har markeret viste krediteringer (punkt 7) kan brugsmønsteret (punkt 8-9) ikke fortsætte.

Tabel 5 Detaljeret brugsmønster - eksportering af kreditering

<p>Brugsmønster: Opdatere program</p>
--

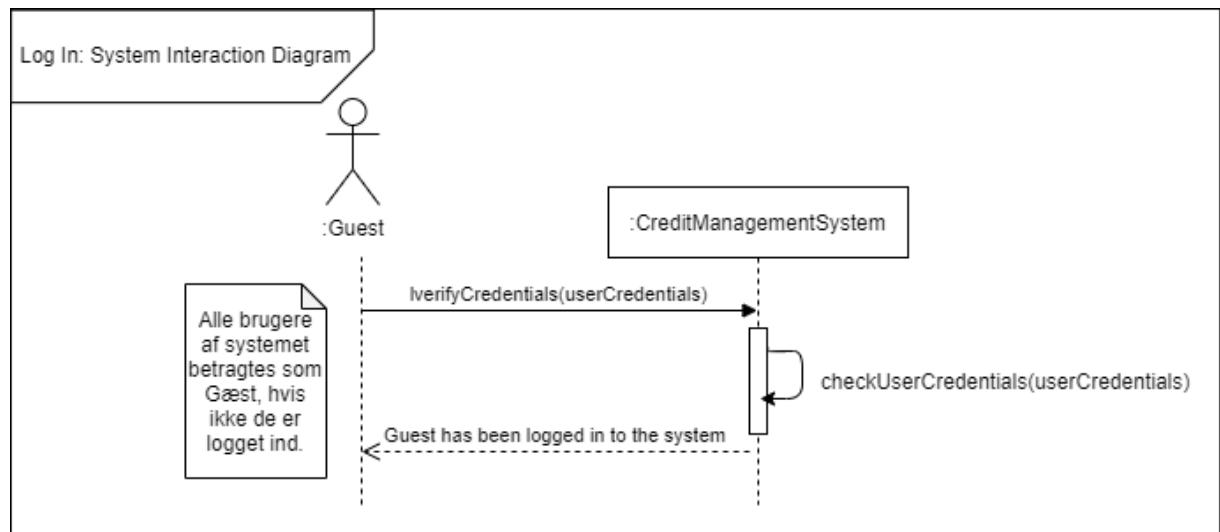
ID: B17
Primære aktører: Systemadministrator & producer omtales aktør fremover
Sekundære aktører: Ingen
Kort beskrivelse: Systemadministratoren skal kunne opdatere programmer, samt den producer som har oprettet programmet
Prækonditioner (Pre-conditions): <ul style="list-style-type: none">• Aktøren skal være logget ind• Program skal eksistere i systemet
Hovedhændelsesforløb (main flow): <ol style="list-style-type: none">1. Brugsmønstret starter, når systemadministratoren gerne vil opdatere et program.2. Aktøren trykker på knappen opdater ved det givne program.<ol style="list-style-type: none">2.1. Hvis aktøren er en producer, er det kun muligt at trykke på opdater-knappen, hvis det er den samme, som har oprettet programmet3. Systemet synliggør opdateringsmuligheder vedrørende programoplysninger.<ol style="list-style-type: none">3.1. Hvis systemadministrator ønsker at opdatere krediteringer, sker følgende:<ol style="list-style-type: none">3.1.1. B11 Opdater kreditering «extends»4. Systemadministrator udfører ønskede opdateringer5. Systemadministrator trykker på knappen gem opdateringer6. Systemet gemmer udførte opdateringer7. Systemet opretter en systemlog med gældende opdateringsdetaljer.8. Det opdaterede program sættes til at skulle godkendes igen<ol style="list-style-type: none">8.1. Programmet skal derefter igennem B13 Godkend Program igen
Postkonditioner (Post conditions): <ul style="list-style-type: none">• Programoplysningerne er blevet opdateret i systemet• En systemlog er blevet oprettet med de gældende opdateringsdetaljer.
Alternative hændelsesforløb (Alternative flow): <ul style="list-style-type: none">• Aktøren kan afbryde brugsmønstret overalt fra punkt 2-6
Krydsreferencer: <ul style="list-style-type: none">• B11 Opdater kreditering• B13 Godkend Program

Tabel 6 Detaljeret brugsmønster - opdatere program

C Brugsmønsterrealiseringer

Brugsmønsterrealiseringer af brugsmønstrene B04 Log-ind, B06 Log-ud, B10 Opret program og B13 Godkend krediteringer.

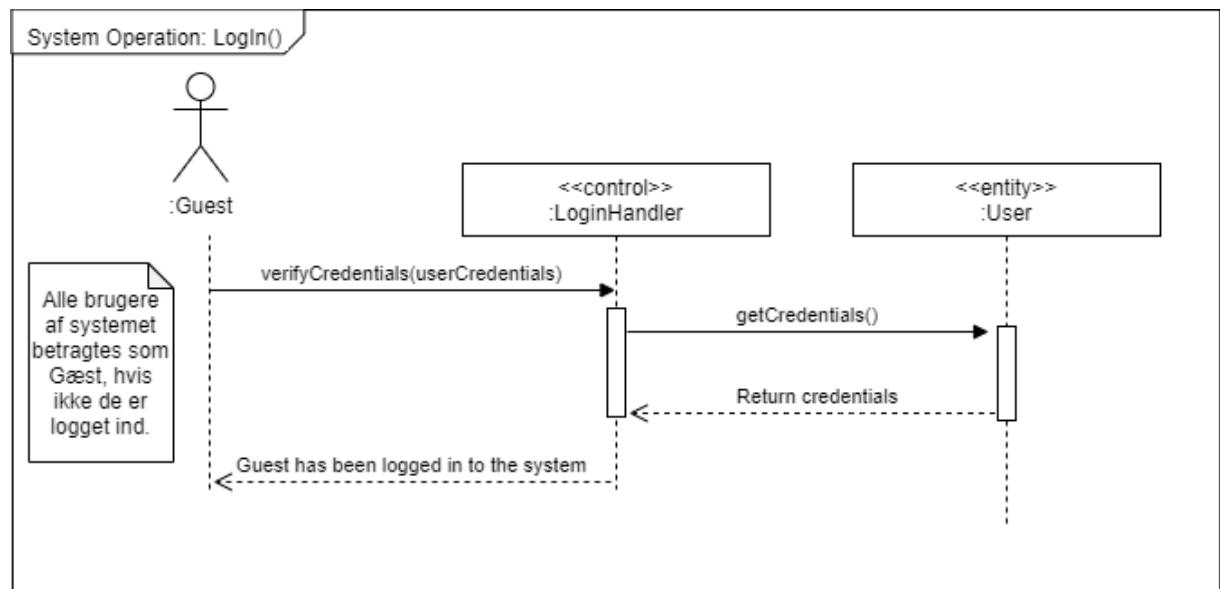
Brugsmønsterrealisering af B04 Log-ind



Figur 2 Systeminteraktionsdiagram - B04 Log-ind

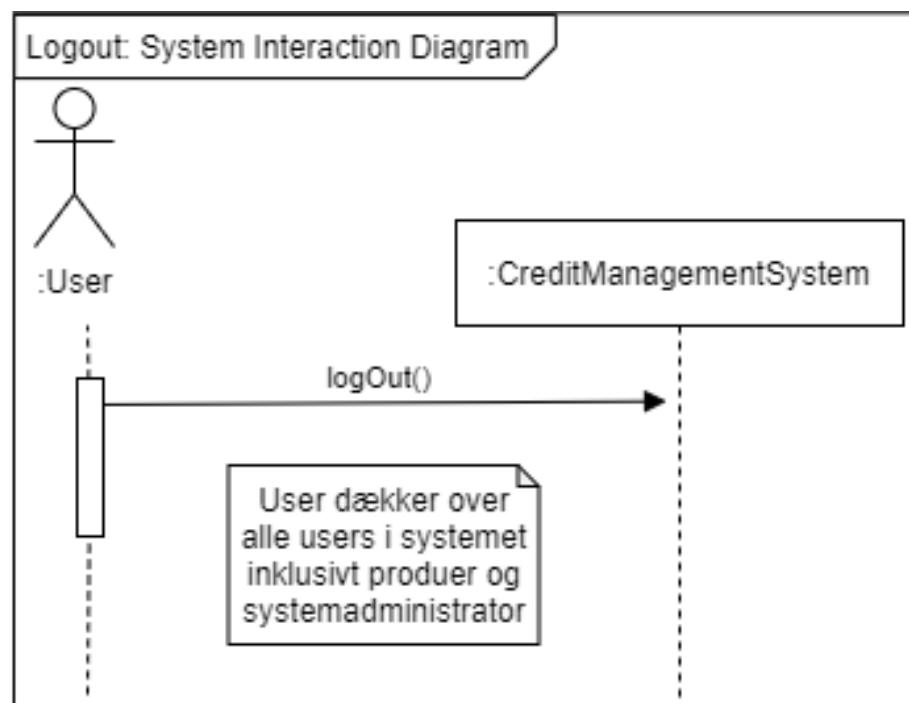
Contract	
Operation:	logIn(userCredentials)
Cross reference:	B04 Log-ind
Responsibility	<ul style="list-style-type: none"> To log-in the specific guest if the given user credentials match the credentials of a user in the system.
Output	<ul style="list-style-type: none"> checkUserCredentials(userCredentials)
Preconditions	A current Guest instance exists.
Postconditions	<p>If the user credentials are valid:</p> <ul style="list-style-type: none"> The Guest was logged in to the system with the associated user credentials.

Tabel 7 Systemoperationskontrakt - B04 Log-ind



Figur 3 Systemoperationssekvensdiagram - B04 Log-ind

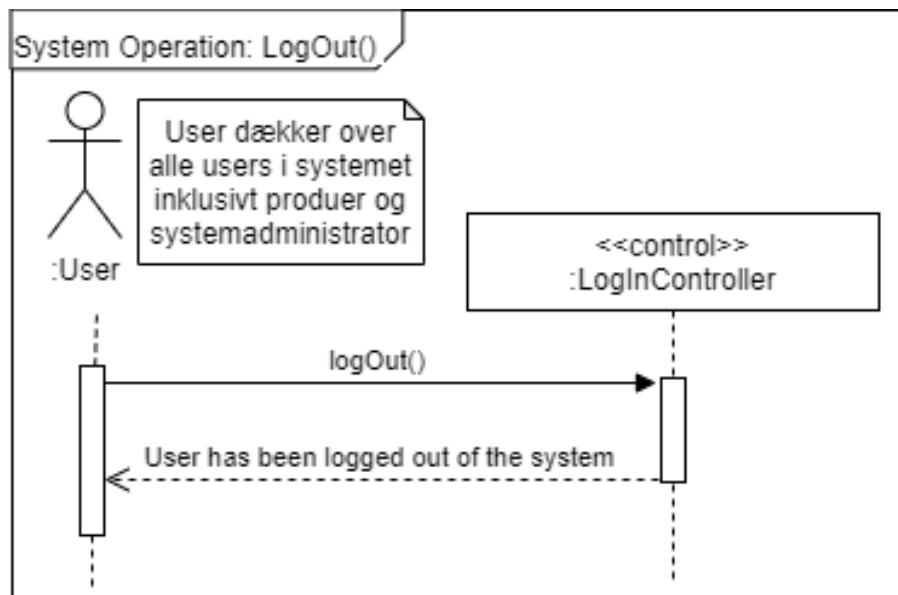
Brugsmønsterrealisering af B06 Log-ud



Figur 4 Systeminteraktionsdiagram - B06 Log-ud

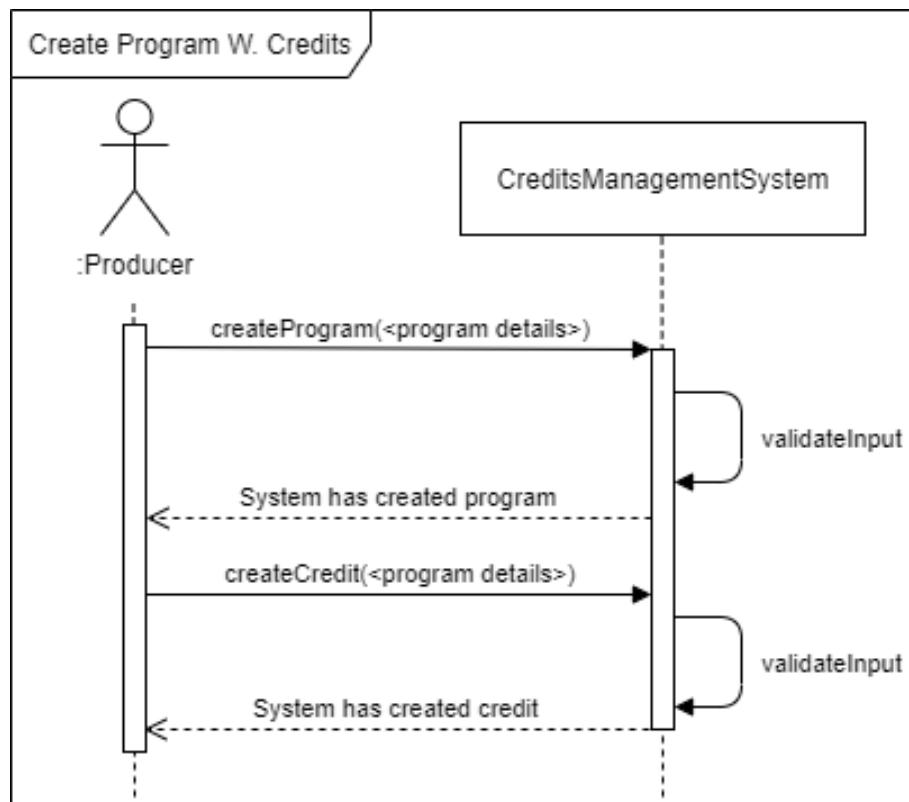
Contract	
Operation:	logOut(userCredentials)
Cross reference:	B06 Log-ud
Responsibility	<ul style="list-style-type: none"> To log out the specific user of the system and return to the standard-page (as a guest).
Output	<ul style="list-style-type: none"> logOutConfirmation()
Preconditions	The current user must be logged in.
Postconditions	<p>If the user confirms the logout:</p> <ul style="list-style-type: none"> The user was logged out of the system and returned to the standard-page (as a guest)

Tabel 8 Systemoperationskontrakt - B06 Log-ud



Figur 5 Systemoperationssekvensdiagram - B06 Log-ud

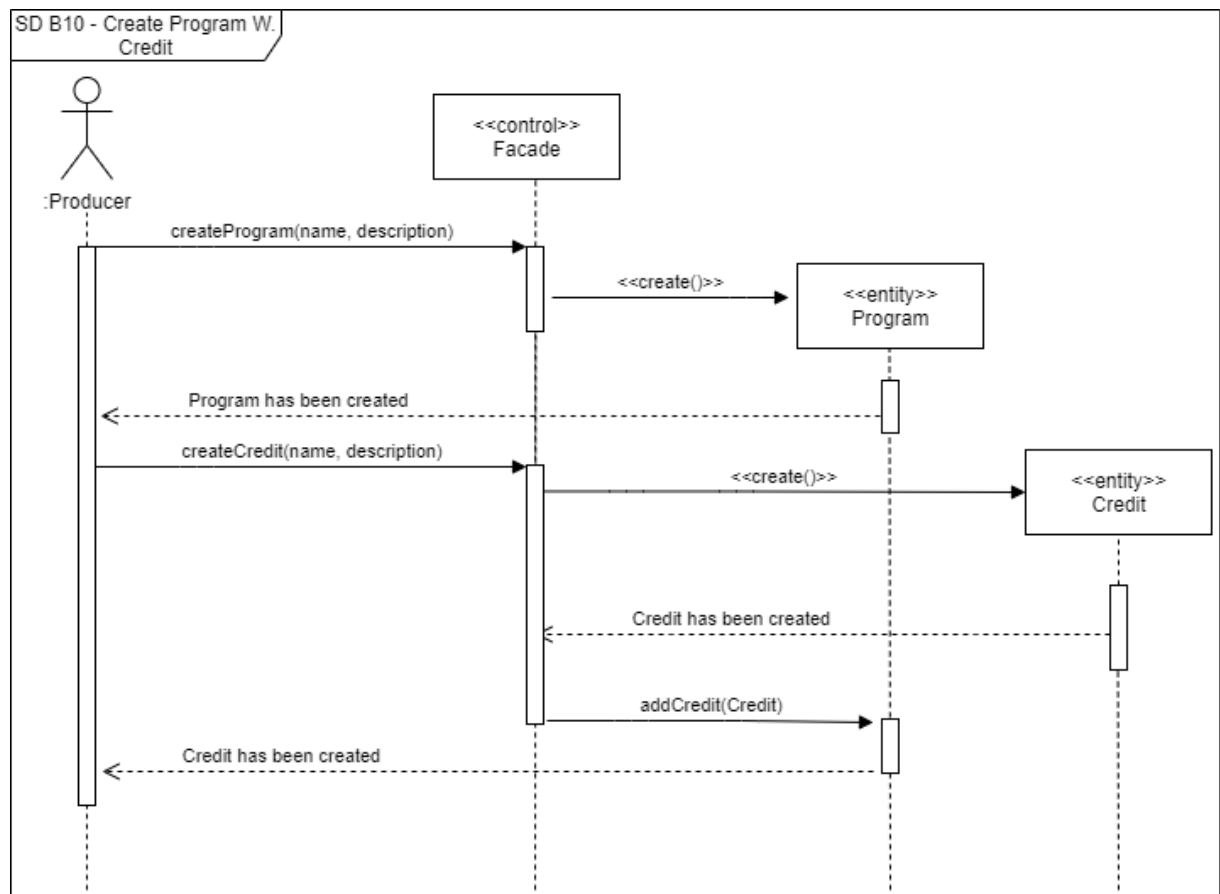
Brugsmønsterrealisering af B10 Opret program



Figur 6 Systeminteraktionsdiagram - B10 Opret program

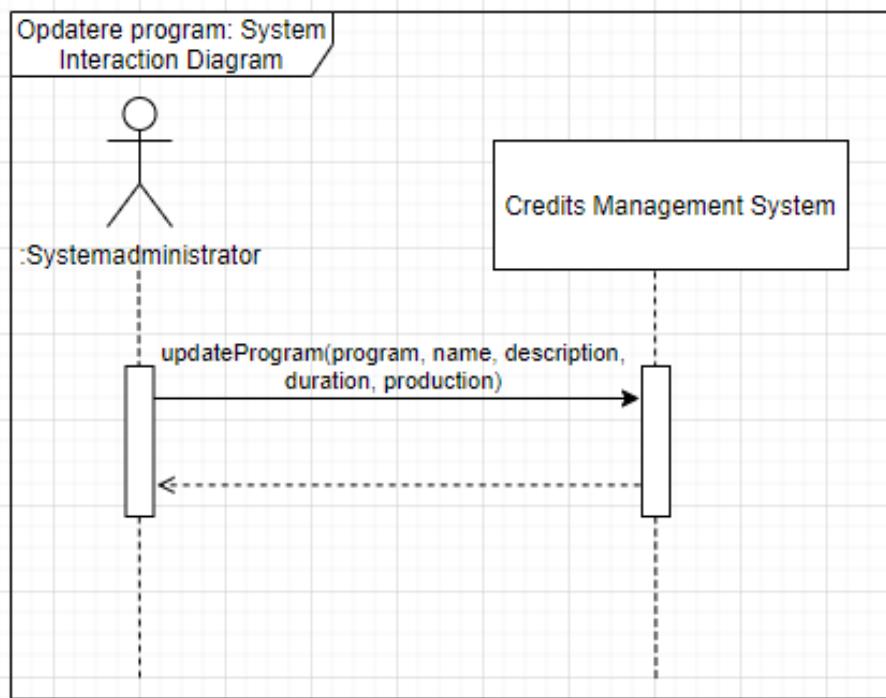
Contract	
Operation:	<code>createProgram(programDetails)</code>
Cross reference:	B10 Opret program
Responsibility	<ul style="list-style-type: none"> To create a TV program with a fitting set of credits.
Output	<ul style="list-style-type: none"> <code>checkForm()</code>
Preconditions	The producer is logged in.
Postconditions	<p>If the form is filled out correctly:</p> <ul style="list-style-type: none"> The system creates a program with the specified credit, which then is visible to the producer.

Tabel 9 Systemoperationskontrakt - B10 Opret program



Figur 7 Systemoperationssekvensdiagram - B10 Opret program

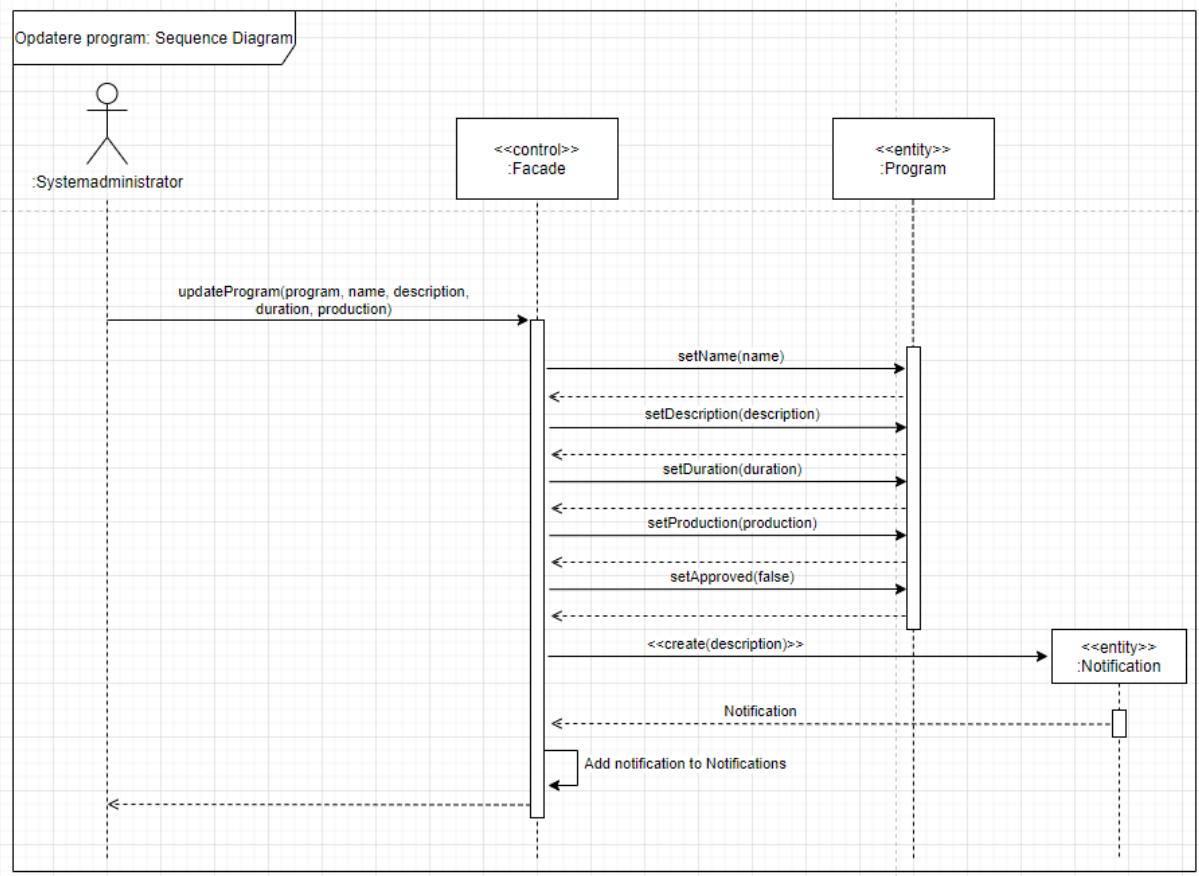
Brugsmønsterrealisering af B17 Opdater program



Figur 8 Systeminteraktionsdiagram - B17 Opdater program

Contract	
Operation:	updateProgram(program, name, description, duration, production)
Cross reference:	B17 Opdater program
Responsibility	<ul style="list-style-type: none"> • At opdatere et program og dets navn, længde, produktion og beskrivelse hvis betingelserne for at opdatere det er mødt. • Et program kan kun opdateres hvis brugeren er programmets producer eller system administrator. • At oprette en notifikation der beskriver ændringerne på programmet.
Output	En notifikation
Preconditions	<ul style="list-style-type: none"> • En instans af en system administrator eller producer eksisterer. • En instans af et program eksisterer.
Postconditions	Hvis betingelserne for at opdatere programmet er sande: <ul style="list-style-type: none"> • Programmet er blevet opdateret • Der er blevet oprettet en Notifikation i systemloggen

Tabel 10 Systemoperationskontrakt - B17 Opdater program



Figur 9 Systemoperationssekvensdiagram - B17 Opdater program

D Fuldt SQL-Script

```

1  DROP TABLE IF EXISTS programs CASCADE;
2  DROP TABLE IF EXISTS productions CASCADE;
3  DROP TABLE IF EXISTS users CASCADE;
4  DROP TABLE IF EXISTS transmissions CASCADE;
5  DROP TABLE IF EXISTS episodes CASCADE;
6  DROP TABLE IF EXISTS credits CASCADE;
7  DROP TABLE IF EXISTS tv_series CASCADE;
8  DROP TABLE IF EXISTS admins CASCADE;
9  DROP TABLE IF EXISTS producers CASCADE;
10  DROP TABLE IF EXISTS creditedPeople CASCADE;
11  DROP TABLE IF EXISTS notifications CASCADE;
12
13  BEGIN;
  
```

```
14 CREATE TABLE creditedPeople(
15     id UUID PRIMARY KEY,
16     name VARCHAR(128) NOT NULL
17 );
18
19 CREATE TABLE productions(
20     id SERIAL PRIMARY KEY,
21     name VARCHAR(128) NOT NULL
22 );
23
24 CREATE TABLE users(
25     id UUID PRIMARY KEY,
26     username VARCHAR(20) NOT NULL,
27     password INTEGER NOT NULL,
28     type VARCHAR(128) NOT NULL
29 );
30
31 CREATE TABLE programs(
32     id UUID PRIMARY KEY,
33     name VARCHAR(128) NOT NULL,
34     description VARCHAR(1024) NOT NULL,
35     createdById UUID NOT NULL REFERENCES users(id),
36     duration INTEGER NOT NULL,
37     approved BOOLEAN NOT NULL,
38     production INTEGER NOT NULL REFERENCES productions(id)
39 );
40
41 CREATE TABLE transmissions(
42     programsId UUID PRIMARY KEY REFERENCES programs(id)
43 );
44
45 CREATE TABLE episodes(
46     programsId UUID PRIMARY KEY REFERENCES programs(id),
47     tvSeriesId UUID NOT NULL,
48     episodeNo INTEGER NOT NULL,
49     seasonNo INTEGER NOT NULL
50 );
51
52 CREATE TABLE credits(
53     creditedPersonId UUID NOT NULL REFERENCES creditedPeople(id),
54     role VARCHAR(128) NOT NULL,
```

```
55     programId UUID NOT NULL,  
56     PRIMARY KEY (creditedPersonId, role, programId)  
57 );  
58  
59 CREATE TABLE tv_series(  
60     id UUID PRIMARY KEY,  
61     name VARCHAR(128) NOT NULL,  
62     description VARCHAR(1024),  
63     createdByUuid UUID NOT NULL REFERENCES users(id)  
64 );  
65  
66 CREATE TABLE notifications(  
67     id SERIAL PRIMARY KEY,  
68     title VARCHAR(1024) UNIQUE NOT NULL,  
69     seen BOOLEAN NOT NULL  
70 );  
71  
72 COMMIT;  
73  
74 CREATE VIEW transmissionsView AS  
75 SELECT id, name, description, createdById, duration, approved,  
    production FROM programs, transmissions  
76 WHERE programs.id = transmissions.programsId;  
77  
78 CREATE VIEW episodesView AS  
79 SELECT id, tvSeriesId, name, description, createdById, episodeNo,  
    seasonNo, duration, approved, production FROM programs,  
    episodes  
80 WHERE programs.id = episodes.programsId;  
81  
82 BEGIN;  
83 --  
84 INSERT INTO creditedPeople (id, name) VALUES  
85 ('05a639ea-79a3-43b9-b0d7-079b6fce3e56', 'Mathias Engmark'),  
86 ('af8c0e36-3abf-4f97-99b6-3d7c6d94ce1b', 'anton Irvold'),  
87 ('260f65f3-4110-4fd9-b21e-610fdb18a343', 'Christoffer Krath'),  
88 ('6cd868ae-2666-408b-8d06-5945bf742d6c', 'Nicklas Jensen'),  
89 ('1e3ccd36-f8d8-40fd-bd88-37b2843c8b72', 'Kasper Stokholm'),  
90 ('2685807d-a300-46fa-b087-8abd995befa6', 'Oliver Heine');  
91  
92 --
```

```
93 INSERT INTO productions(name) VALUES
94 ('TV2'),
95 ('Nordisk Film');
96
97 --
98 INSERT INTO users (id, username, password, type) VALUES
99 ('5df26920-3da5-4b07-a394-641306d68561', 'kapper', '-1138619133',
  'Producer'),
100 ('b632b093-dc71-413d-a73e-d672b398c88c', 'producer', '-1003761774',
   'Producer'),
101 ('66789049-a663-4a07-bcd1-d4ac7e481a20', 'admin', '92668751',
  'admin');

102 --
103 --
104 INSERT INTO programs (id, name, description, createdBy, duration
  , approved, production) VALUES
105 ('f85a5a3a-dd0e-4943-a54c-940ead342eb', 'Everything burns', 'a
  hous burned to the ground', '5df26920-3da5-4b07-a394-641306
  d68561', 1, true, 1),
106 ('0ee8bea9-f0f6-41dc-8658-e0e8475553cc', 'De sproejter alle sammen
  ', 'alle hold skal bage en lagkage, og vaere faellels om at
  sproejte kagecrem rund paa kagen', 'b632b093-dc71-413d-a73e-
  d672b398c88c', 1, false, 2),
107 ('2a93e924-6638-465a-b5cd-cb8d16f00d69', 'Top Gun', 'En skyde film
  ', 'b632b093-dc71-413d-a73e-d672b398c88c', 124, true, 1),
108 ('ae880937-b2cd-4206-b8c8-35966165390a', 'Melodi Gran Prix', 'Et
  sang show', '5df26920-3da5-4b07-a394-641306d68561', 12, false,
  2),
109 ('3129a54d-2d5f-4fbc-a39a-9c478905ff11', 'Flyvende Farmor', 'En
  overskudsfarmor paa tur', 'b632b093-dc71-413d-a73e-d672b398c88c
  ', 234, true, 1);

110 --
111 --
112 INSERT INTO transmissions (programsId) VALUES
113 ('2a93e924-6638-465a-b5cd-cb8d16f00d69'),
114 ('ae880937-b2cd-4206-b8c8-35966165390a'),
115 ('3129a54d-2d5f-4fbc-a39a-9c478905ff11');

116 --
117 --
118 INSERT INTO tv_series (id, name, description, createdBy) VALUES
```

```
119 ('0c7bb0e9-7adb-4f3f-a8fd-6d7609c1f6cc', 'Chicago Fire', 'En serie  
om CPD', '5df26920-3da5-4b07-a394-641306d68561'),  
120 ('b3794996-72f6-42fa-90db-1fff32010af3', 'Den Store Bagedyst', 'Du  
ser folk bage - og bliver vildt sulten', 'b632b093-dc71-413d-  
a73e-d672b398c88c');  
121 --  
122 --  
123 INSERT INTO episodes (programsId, tvSeriesId, episodeNo, seasonNo)  
VALUES  
124 ('f85a5a3a-dd0e-4943-a54c-940eed342eb', '0c7bb0e9-7adb-4f3f-a8fd  
-6d7609c1f6cc', 1, 2),  
125 ('0ee8bea9-f0f6-41dc-8658-e0e8475553cc', 'b3794996-72f6-42fa-90db  
-1fff32010af3', 2, 3);  
126 --  
127 --  
128 INSERT INTO credits (creditedPersonId, role, programId) VALUES  
129 ('05a639ea-79a3-43b9-b0d7-079b6fce3e56', 'Casting', '2a93e924  
-6638-465a-b5cd-cb8d16f00d69'),  
130 ('260f65f3-4110-4fd9-b21e-610fdb18a343', 'Billed- og lydredigering  
, '2a93e924-6638-465a-b5cd-cb8d16f00d69'),  
131 ('af8c0e36-3abf-4f97-99b6-3d7c6d94ce1b', 'Kor', 'ae880937-b2cd  
-4206-b8c8-35966165390a'),  
132 ('2685807d-a300-46fa-b087-8abd995befa6', 'Efter ide af', '3129a54d  
-2d5f-4fbca39a-9c478905ff11'),  
133 ('6cd868ae-2666-408b-8d06-5945bf742d6c', 'Dronefoerer', 'f85a5a3a-  
dd0e-4943-a54c-940eed342eb'),  
134 ('1e3ccd36-f8d8-40fd-bd88-37b2843c8b72', 'Danske undertekster', '  
f85a5a3a-dd0e-4943-a54c-940eed342eb');  
135  
136  
137 INSERT INTO notifications (title, seen) VALUES  
138 ('Kapper created a person with the name Mathias Engmark on  
13-04-2021 13:45:23', true),  
139 ('Kapper created a person with the name anton Irvold on 13-04-2021  
13:55:23', true),  
140 ('Producer created a person with the name Christoffer Krath on  
13-04-2021 14:04:39', true),  
141 ('Kapper created a person with the name Nicklas Jensen on  
14-04-2021 11:55:03', true),  
142 ('Producer created a person with the name Kasper Stokholm on  
14-04-2021 12:13:30', true),
```

```
143 ('Kapper created a person with the name Oliver Heine on 14-04-2021  
144     12:15:23', true),  
144 ('Producer created a TV-series with the name Chicago Fire on  
145     14-04-2021 13:25:23', true),  
145 ('Kapper created an episode with the title Everything burns on  
146     14-04-2021 13:27:48', true),  
146 ('Kapper created a TV-series with the name Den Store Bagedyst on  
147     15-04-2021 11:45:46', true),  
147 ('Producer created an episode with the title De sproejter alle  
148     sammen on 15-04-2021 12:00:12', true),  
148 ('producer created a transmission with the title Top Gun on  
149     16-04-2021 14:23:56', true),  
149 ('Kapper created a transmission with the title Melodi Gran Prix on  
150     16-04-2021 21:34:42', true),  
150 ('Producer created a transmission with the title Flyvende Farmor  
151     on 17-04-2021 23:21:34', true),  
151 ('Producer added Mathias Engmark as a credit to the program Top  
152     Gun with the role Casting on 18-04-2021 08:32:45', true),  
152 ('Producer added Christoffer Krath as a credit to the program Top  
153     Gun with the role Billed- og lydredigering on 18-04-2021  
153     08:33:00', true),  
153 ('Kapper added anton Irvold as a credit to the program Melodi Gran  
154     Prix with the role Kor on 18-04-2021 08:34:45', true),  
154 ('Producer added Oliver Heine as a credit to the program Flyvende  
155     Farmor with the role Efter ide af on 18-04-2021 08:42:45', true  
155     ),  
155 ('Kapper added Nicklas Jensen as a credit to the program Top Gun  
156     with the role Dronefoere on 18-04-2021 09:32:45', true),  
156 ('Kapper added Kasper stokhold as a credit to the program Top Gun  
157     with the role Danske undtekster on 18-04-2021 10:33:00', true  
157     ),  
157 ('admin approved a program with the UUID f85a5a3a-dd0e-4943-a54c  
158     -940eead342eb and name Everything burns on 20-04-2021 12:34:21',  
158     false),  
158 ('admin approved a program with the UUID 2a93e924-6638-465a-b5cd-  
159     cb8d16f00d69 and name Top Gun on 20-04-2021 12:34:21', false),  
159 ('admin approved a program with the UUID 3129a54d-2d5f-4fbc-a39a-9  
159     c478905ff11 and name Flyvende Farmor on 20-04-2021 12:34:21',  
159     false);  
160 COMMIT;
```

Listing 1: Systemets fulde SQL-Script

E Brugertest: Feedback

Feedback fra testperson 2

Tabel 11 Feedback fra person 2

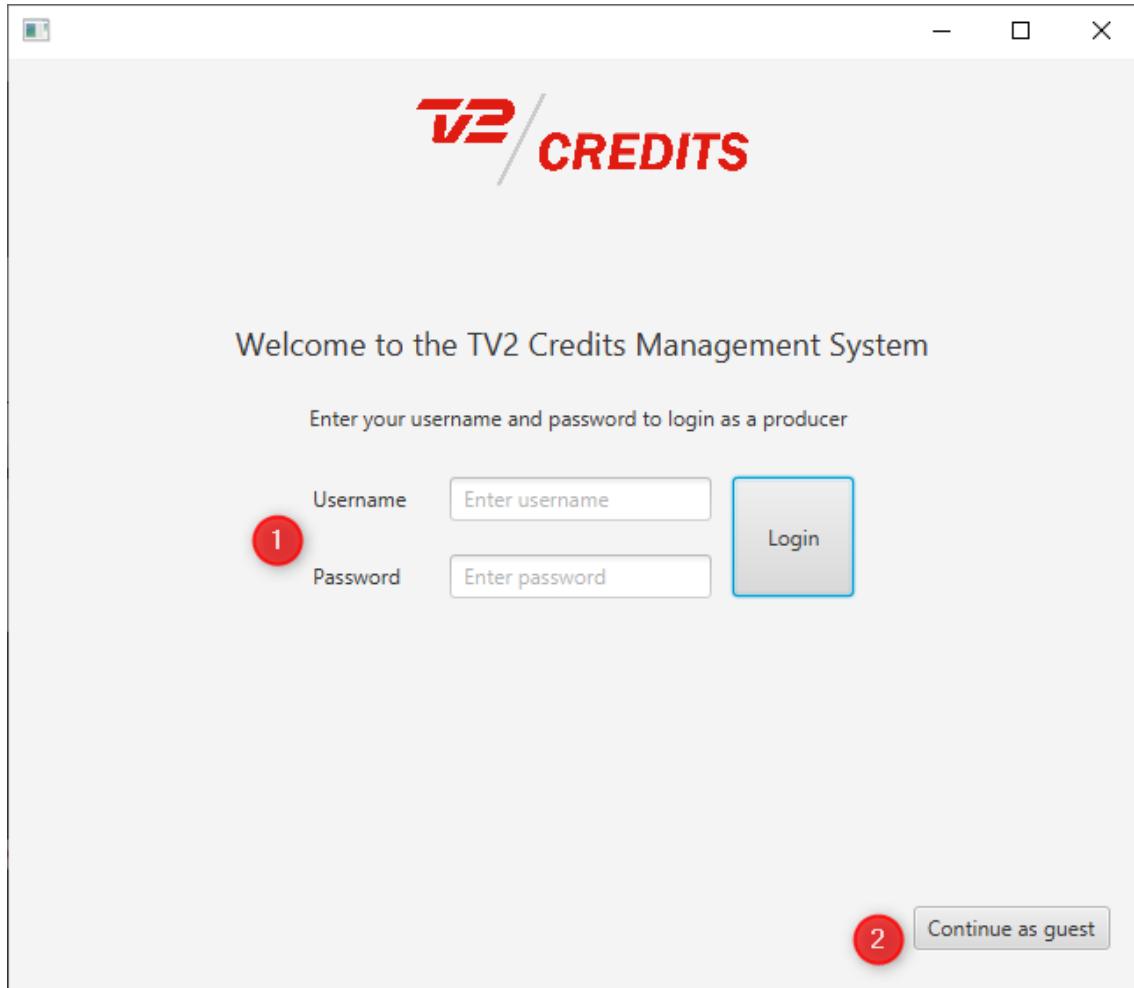
Spørgsmål	Svar
Hvordan var din oplevelse med at logge ind som hhv. gæst og producer/system administrator?	Nemt
Hvordan var din oplevelse med at søge og vise programmer og deres krediteringer som gæst? Er informationen præsenteret ordentligt? Hvad kunne gøres for at forbedre dette?	Nemt
Hvordan var din oplevelse med at oprette et program og krediteringer til programmet? Er brugergrænsefladen forståelig og tilfredsstillende intuitiv? Er det overskueligt og pænt? Har du oplevet nogle fejl eller mangler? Hvordan er din forståelse af hvad programtyperne ville være i den virkelig verden? Giv gerne et eksempel på hvad de hver kunne være (transmission, episode og tv-serie)	Forholdsvis nemt. Brugergrænsefladen er forståelig, simpelt. Massere af fejl, er skrevet ned. Transmission? Episode er enkelte dele af en tv-serie.
Hvordan var din oplevelse med at opdatere programmer? Oplevede du nogle fejl eller mangler? Var det en intuitiv oplevelse?	Det var meget nemt.
Hvordan var din oplevelse med at godkende programmer som system administrator? Hvordan er din forståelse af hvad der sker, når man godkender et program?	Nemt igen, når noget godkendes sidder der en højere i hierarkiet og efterser rettelserne.

Har du andre bemærkninger?	Søge felter passer ikke til størrelsen af vinduet. "Tab rækkefølgen" i "Create" er forkert. Uklart om der skal stå "minutter" efter Duration. Ikke noget feedback på "Create person" knappen. Search/view opdaterer ikke automatisk.
----------------------------	---

F Brugervejledning

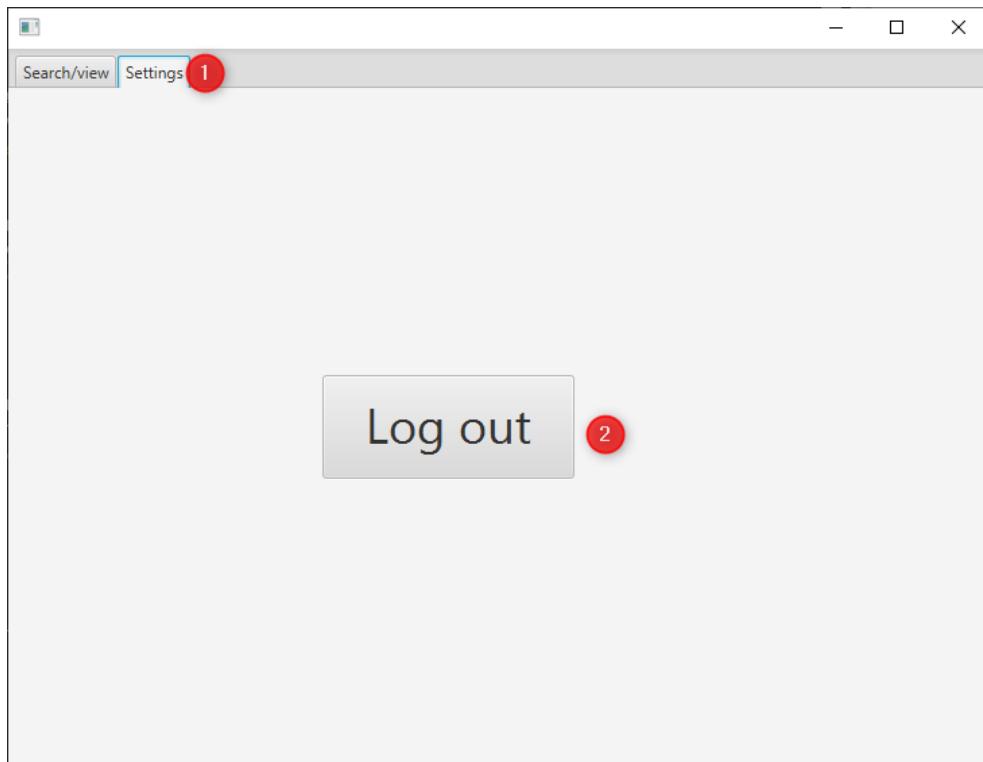
Dette vil være en komplet brugervejledning for henholdsvis gensidige brugere af dette Credits Management System samt producere og TV2s systemadministratorere. I første afsnit, bliver brugerens funktionalitet gennemgået efterfulgt at producerens og systemadministratorens i denne rækkefølge.

Når programmet opstartes, vil der være mulighed for at logge ind. Dette er udelukket for producere samt systemadministratorere. For at logge ind som producer eller systemadministrator, indtast brugernavn samt adgangskode i delterne markeret ved 1. Se Figur 10 For at bruge programmet som almindelig bruger/gæst, tryk på knappen: "Continue as guest", som er placeret nederst i højre hjørne, ved 2. Se Figur 10.



Figur 10 Bruger login.

For at logge ud af systemet, tryk på panelet "Settings", markeret ved 1. Tryk dernæst på knappen "Log Out", markeret ved 2. Se Figur 11.

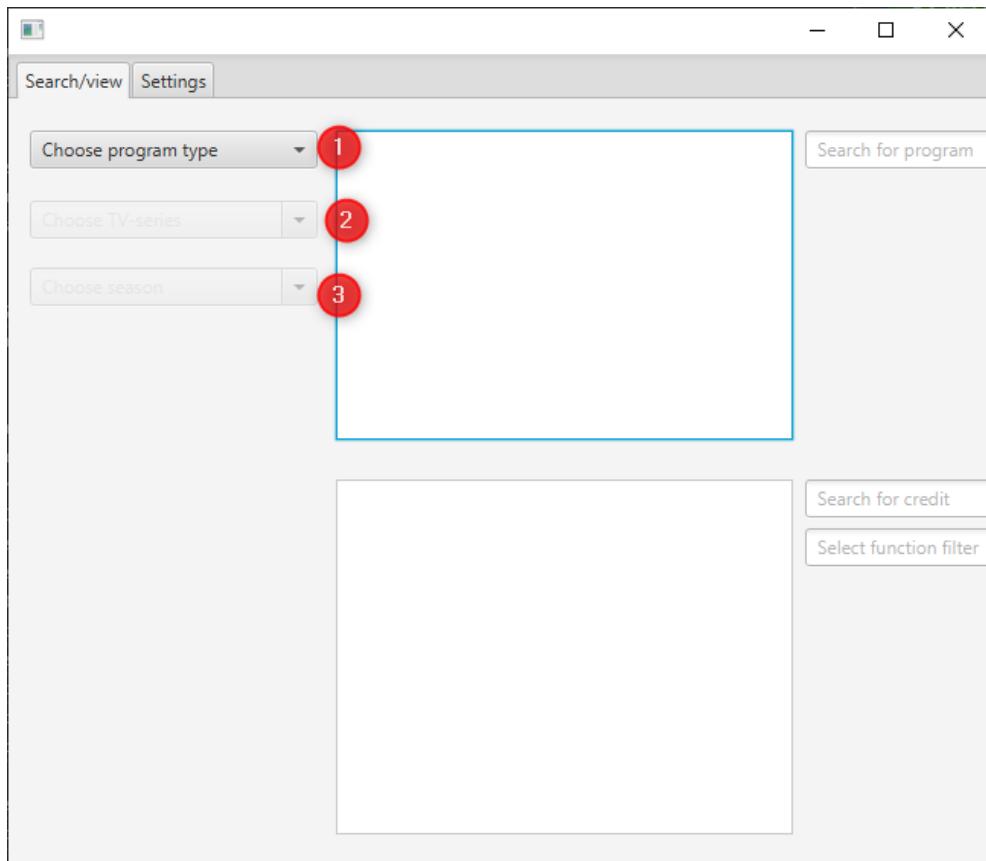


Figur 11 Log ud af programmet.

Vejledning for bruger

I dette program har du muligheden for at gennemsøge krediteringer på programmet, produceret af TV2. Disse programmer er opdelt i transmissioner og TV-Serier. Efter man har logget ind som gæst, har du nu mulighed for at vælge programtype: Transmission og TVSerie. Se Figur 12.

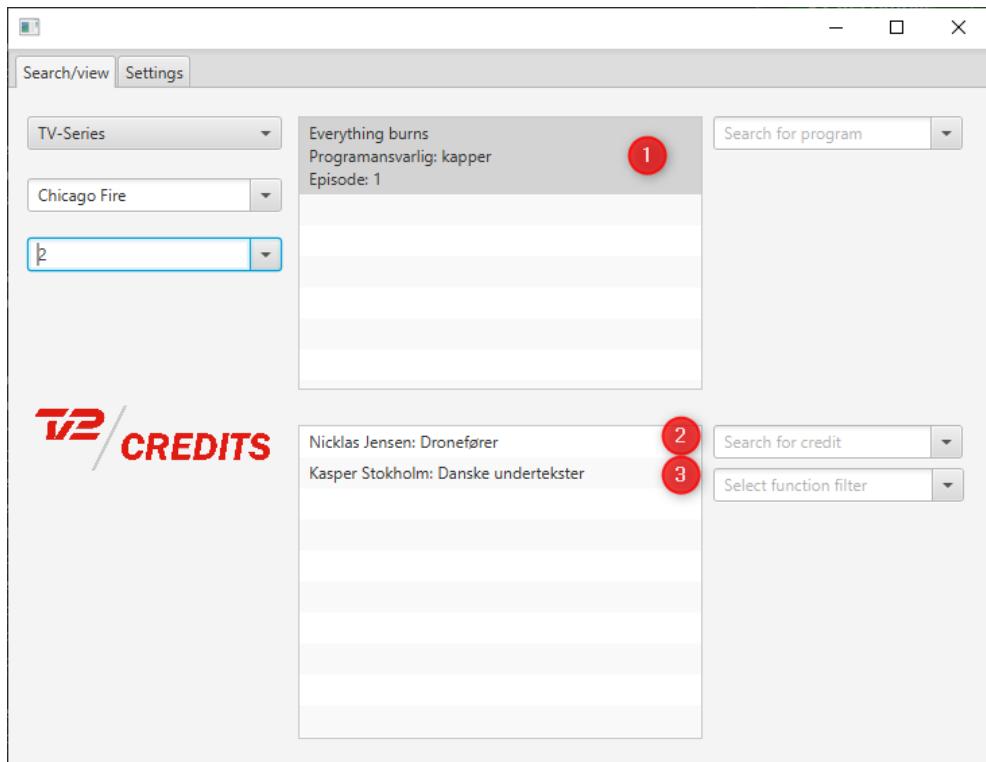
1. Vælg programtype.
2. Hvis programtypen TVSerie er valgt, angiv TVSerien.
3. Angiv sæson for TVSerien.



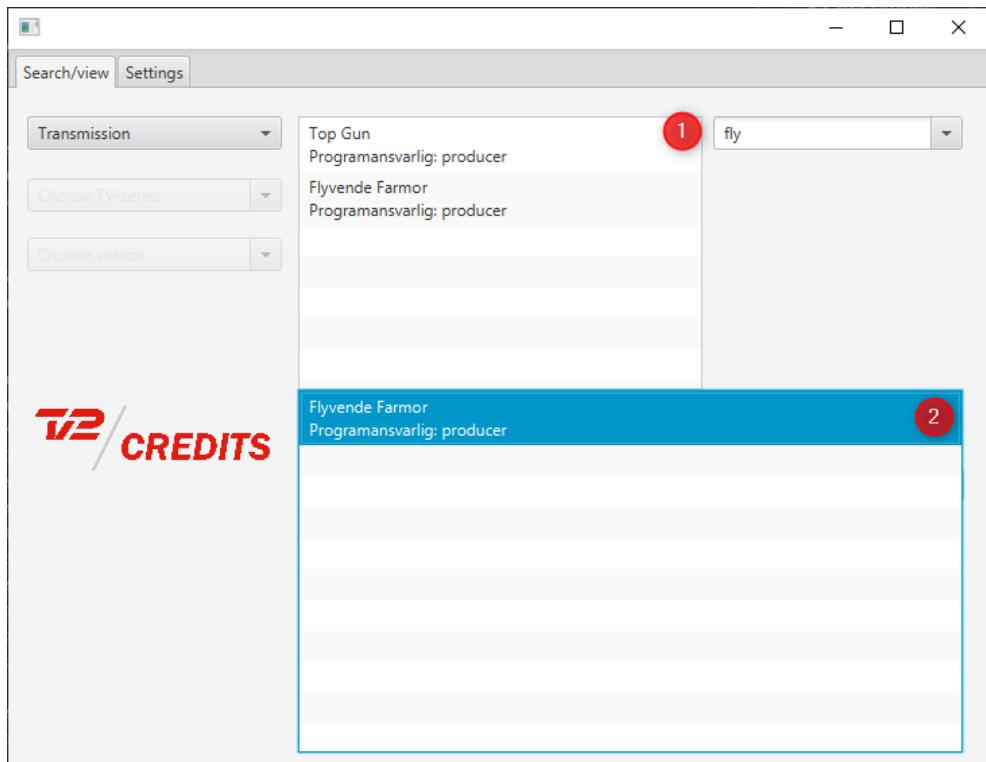
Figur 12 Valg af programtype.

For at se krediteringerne for et pågældende program, trykkes der på episode nummeret/transmissionen markeret ved punkt 1. Herefter kan krediteringerne vælges ved punkt 2 og 3. Se Figur 13. For ordinære transmissioner, se Figur 14.

Appendices



Figur 13 Find kreditering på TVSerie



Figur 14 Find kreditering på transmission

Vejledning for producer

For at logge ind som producer, skal legitimationsoplysninger i felterne markeret ved 1. Se Figur 10.

For at oprette et program gøres følgende: - Se Figur 15.

1. Angiv programtype. (Transmission/TVSerie/Episode)
2. Indskriv navnet på programmet.
3. Indskriv en beskrivelse.
4. Angiv varighed på programmet.
5. Angiv produktionsselskabet.
- Hvis den angivne programtype IKKE er "Episode, spring punkt 6,7 og 8 over.
6. Angiv sæsonnummer.
7. Angiv episodenummer.
8. Angiv TVSerie. (*Bemærk hvis TVSerien ikke er oprettet, kan episoden ikke tilføjes*)
9. Tryk "Create program".

The screenshot shows a software window titled "Producer view". At the top, there are tabs: "Create", "Search/view", "Update", "My Programs", and "Settings". The "Create" tab is selected.

Create program

- 1. "Choose program type" dropdown menu.
- 2. "Name" input field.
- 3. "Description" input field.
- 4. "Duration" input field.
- 5. "Production" dropdown menu.
- 6. "Season" input field.
- 7. "Episode" input field.
- 8. "TV-Series" dropdown menu.
- 9. "Create program" button.

Create person

- Name input field.
- "Create person" button.

Create credit

- Program dropdown menu.
- Credited person dropdown menu.
- Function dropdown menu.
- "Create credit" button.

Figur 15 Oprettelse af program.

For at oprette en person til en kreditering, gøres følgende: - Se Figur 16.

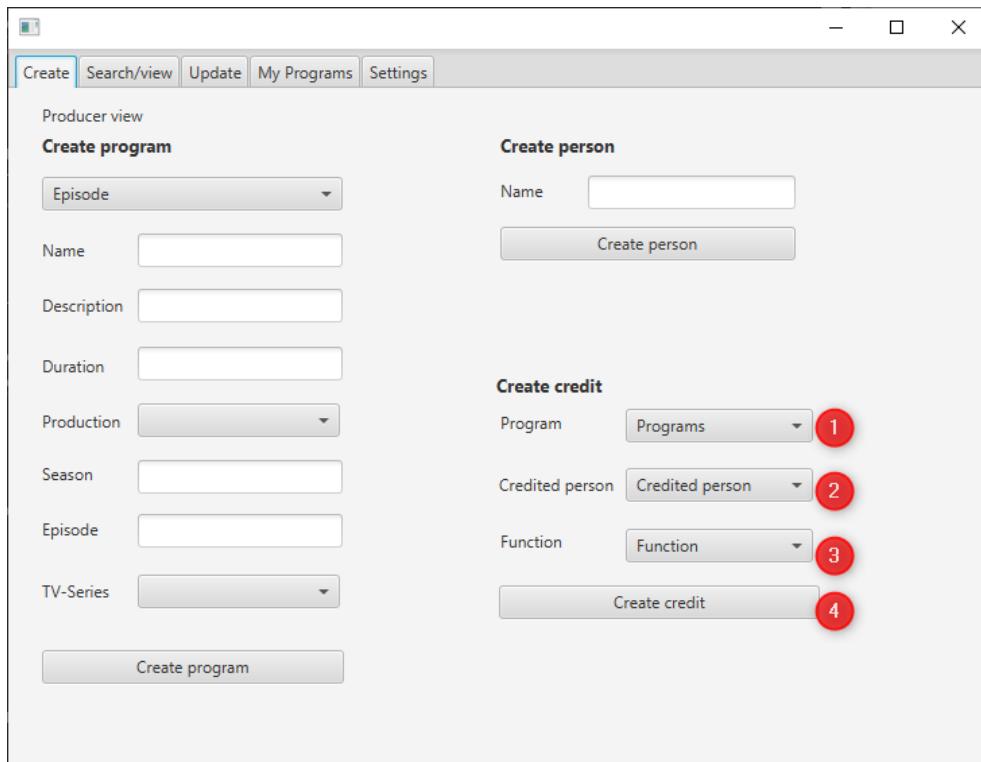
1. Indskriv personens navn.
2. Tryk "Create Person".

The screenshot shows a software window titled 'Producer view'. At the top, there is a navigation bar with tabs: 'Create' (which is selected), 'Search/view', 'Update', 'My Programs', and 'Settings'. Below the navigation bar, there are three main sections: 'Create program' (containing fields for Episode, Name, Description, Duration, Production, Season, Episode, and TV-Series), 'Create person' (containing a 'Name' input field and a 'Create person' button), and 'Create credit' (containing fields for Program, Credited person, and Function). A red circle labeled '1' is drawn around the 'Name' input field in the 'Create person' section. A red circle labeled '2' is drawn around the 'Create person' button in the same section.

Figur 16 Oprettelse af person.

**For at tilknytte en person til en kreditering på et program, gøres følgende: -
Se Figur 17.**

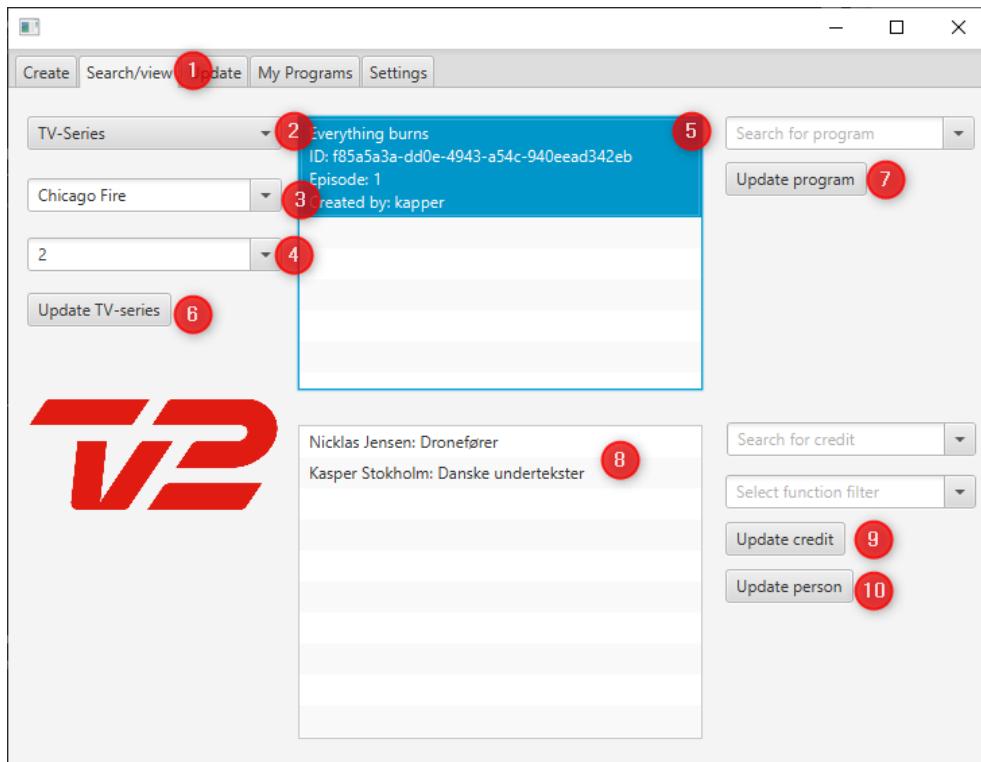
1. Angiv programmet.
2. Angiv personen som skal krediteres.
3. Angiv personens funktionalitet.
4. Tryk "Create Credit".



Figur 17 Oprettelse af kreditering.

**For at gennemse programmer og eventuelt opdatere dem, gøres følgende: -
Se Figur 18.**

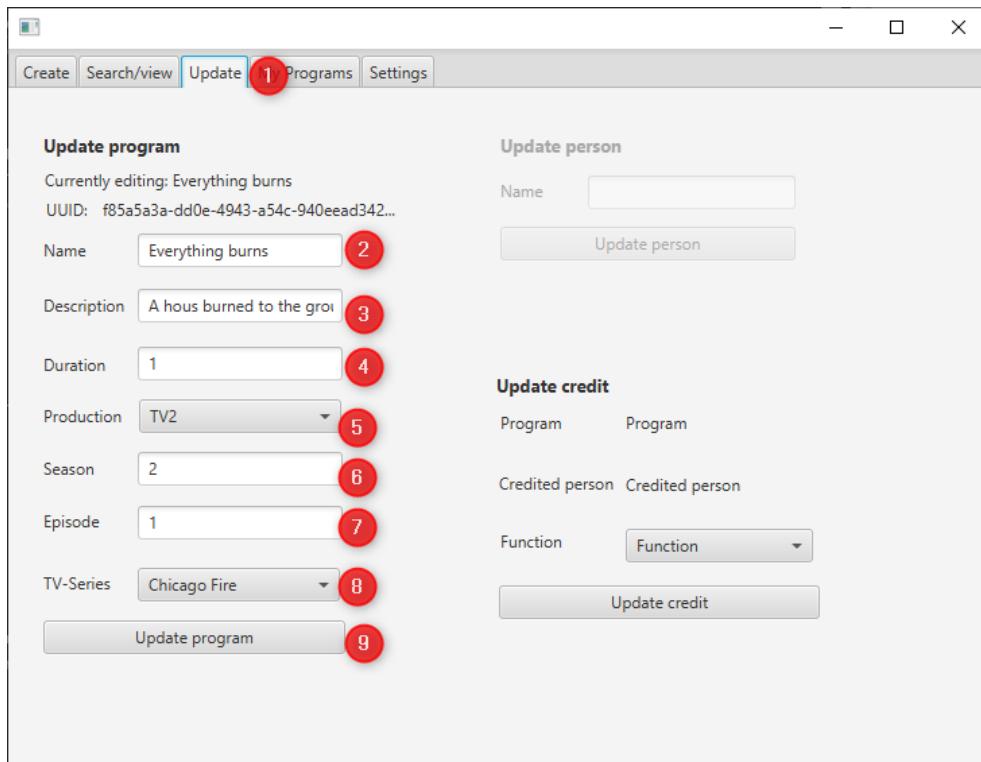
1. Tryk på "Search/view".
2. Angiv programtypen.
- *Hvis den angivne programtype IKKE er "TVSerie", spring punkt 3,4 og 6 over.*
3. Angiv TVSerien.
4. Angiv sæsonnummeret.
5. Marker markerprogrammet.
6. Opdater den angivne TVSerie. Se Figur 20.
7. Opdater det markerede program. Se Figur 19. (*bemærk dette kun er for transmissioner og episoder*)
8. Marker kreditering.
9. Opdater den markerede kreditering. Se Figur 21.
10. Opdater den krediterede person. *Denne funktionalitet er ikke tilgængelig.*



Figur 18 Søgning og opdatering af program samt kreditering.

Ved at have markeret et program og trykke på "Update program", ovenfor kan man nu gøre følgende: - Se Figur 19.

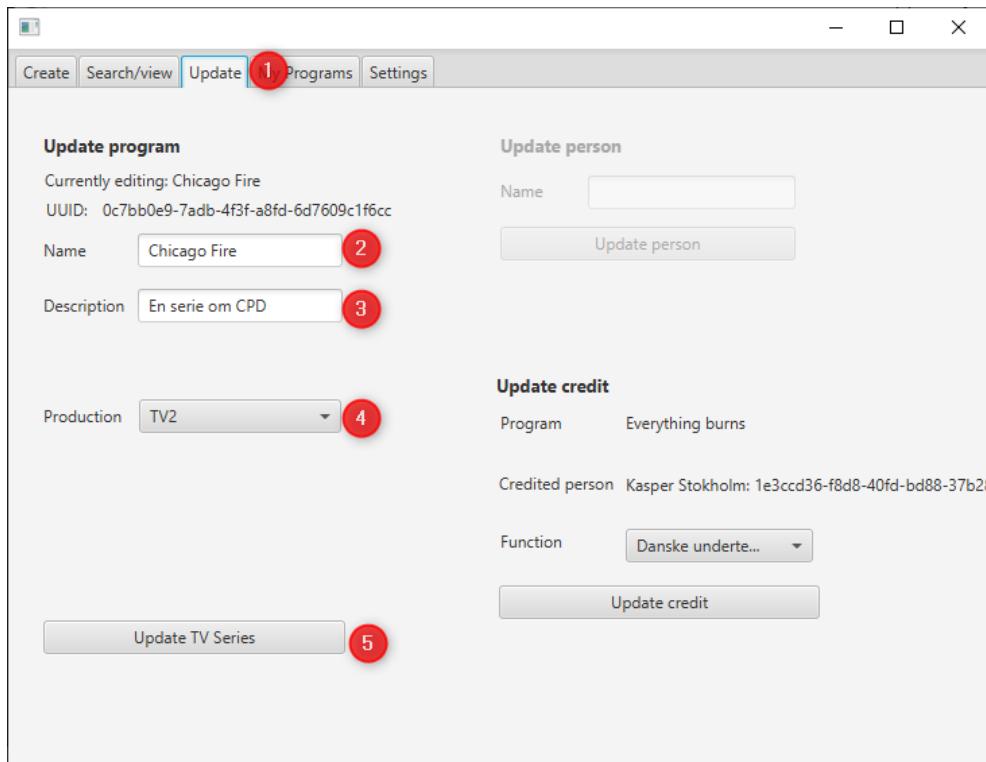
1. Under "Update"-fanen.
2. Angiv navn på programmet.
3. Angiv beskrivelse for programmet.
4. Angiv varighed for programmet.
5. Angiv produktionsselskab.
- Hvis programtypen IKKE er "Episode", spring punkt 6,7 og 8 over.
6. Angiv sæsonnummer.
7. Angiv episodenummer.
8. Angiv TVSerie (Bemærk TVSerien skal være oprettet).
9. Tryk "Update program".



Figur 19 Opdatering af programmer.

Ved at have angivet en TVSerie og trykket "Update TV-series", kan der nu gøres følgende: - Se Figur 20.

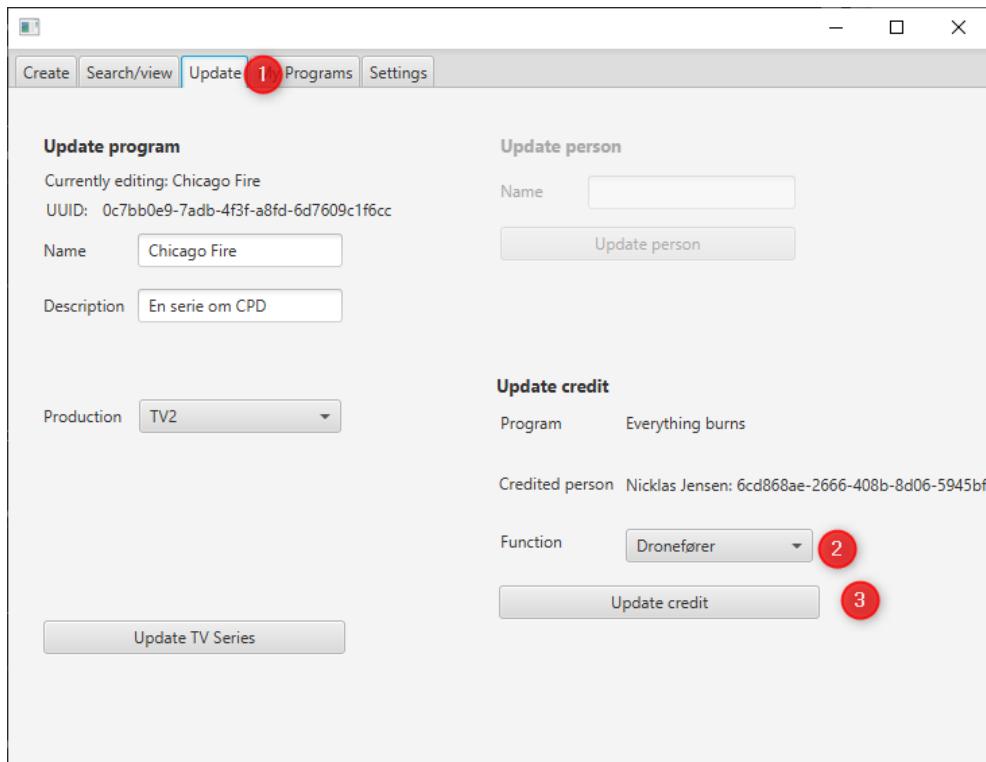
1. Under "Update"-fanen.
2. Angiv navn på programmet.
3. Angiv beskrivelse for programmet.
4. Angiv produktionsselskab.
5. Tryk "Update TV Series".



Figur 20 Opdatering af programmer.

Ved at have markeret en kreditering og trykket "Update credit", kan der nu gøres følgende: - Se Figur 21.

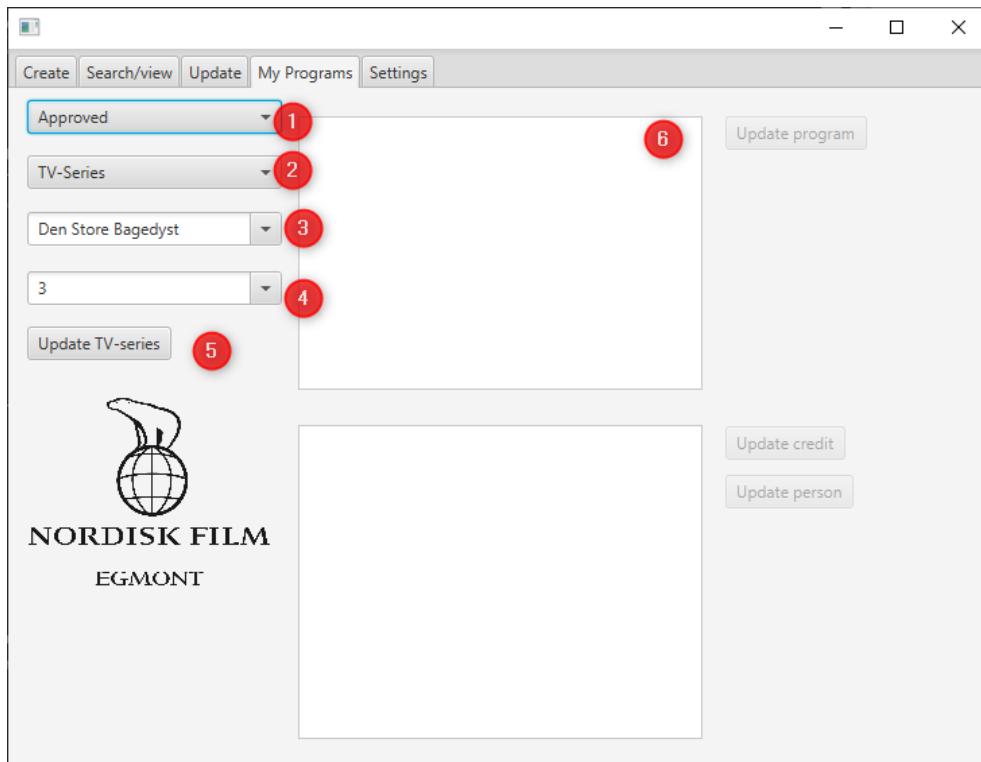
1. Under "Update"-fanen.
2. Angiv funktionen på den krediterede person.
3. Tryk "Update credit".



Figur 21 Opdatering af programmer.

Når et program er oprettet, skal dette godkendes af en systemadministrator.
Man kan derfor se sine egne programmers status herunder. - Se Figur 22.

1. Under "My Programs".
2. Angiv om ens programmer er blevet godkendt, eller afventer godkendelse.
3. Angiv programtype.
 - Hvis programtypen IKKE er "TV-Series", spring punkt 4,5 og 8 over.
4. Angiv TVserie.
5. Angiv sæsonnummer.
6. Herunder er står alle ens oprettede programmer, respektivt til valget under punkt 1.
7. Herunder er alle krediteringer til det tilhørende program indlæst.
8. Opdater den angivne TVserie.
9. Opdater det markerede program, fra punkt 6.
10. Opdater den markrede kreditering, fra punkt 7.
11. Opdater den markerede krediterede person, fra punkt 7. *Denne funktionalitet er ikke tilgængelig.*



Figur 22 Oversigt over egne programmer.

Vejledning for systemadministrator

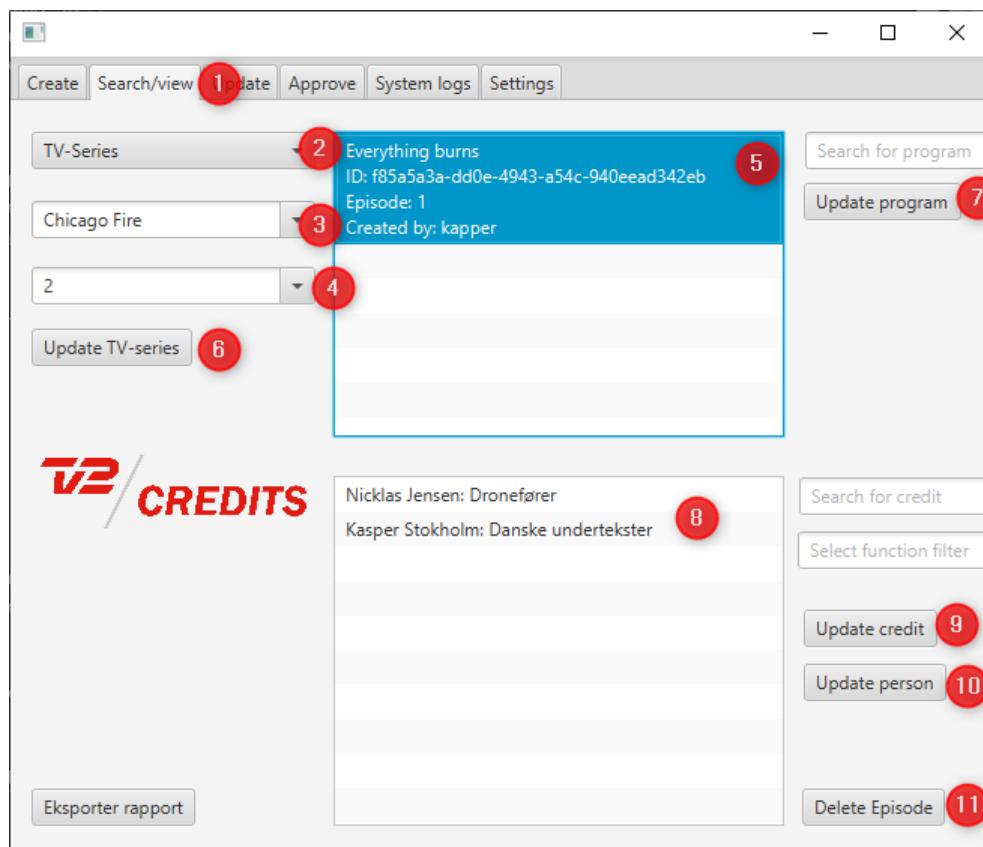
Som systemadministrator har man mange af de samme rettigheder som en producer, og så er par ekstra. Disse uddybes herunder. For vejledning af fælles funktionalitet mellem producer og systemadministrator, bedes man læse Vejledningen for Producer F.

For at gennemse programmer og eventuelt opdatere dem, gøres følgende: -

Se Figur 23.

1. Tryk på "Search/view".
2. Angiv programtypen.
 - *Hvis den angivne programtype IKKE er "TVSerie", spring punkt 3,4 og 6 over.*
3. Angiv TVSerien.
4. Angiv sæsonnummeret.
5. Marker markerprogrammet.
6. Opdater den angivne TVSerie. Se Figur 20.
7. Opdater det markerede program. Se Figur 19. (*bemerk dette kun er for transmissioner og episoder*)
8. Marker kreditering.

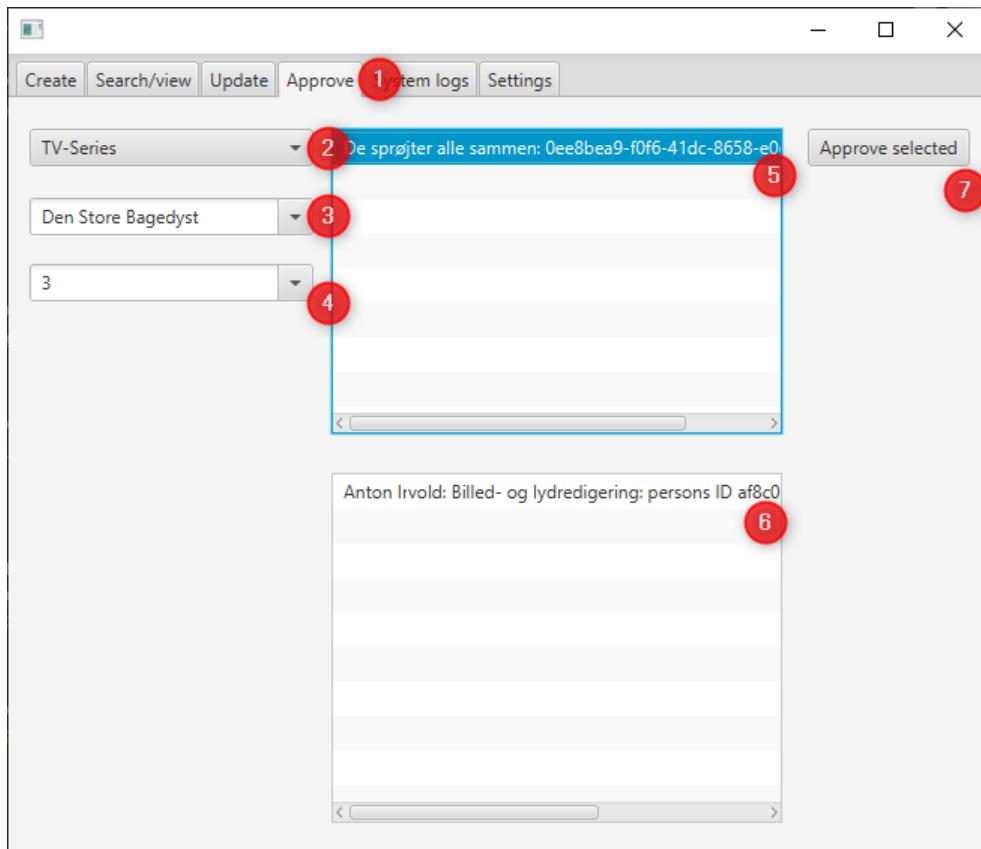
9. Opdater den markerede kreditering. Se Figur 21.
10. Opdater den krediterede person. *Denne funktionalitet er ikke tilgængelig.*
11. Slet det markerede program.



Figur 23 Søgning og opdatering af program samt kreditering.

Som systemadministrator er det ens opgave at godkende programmer og deres krediteringer. Dette gøres således: - Se Figur 24.

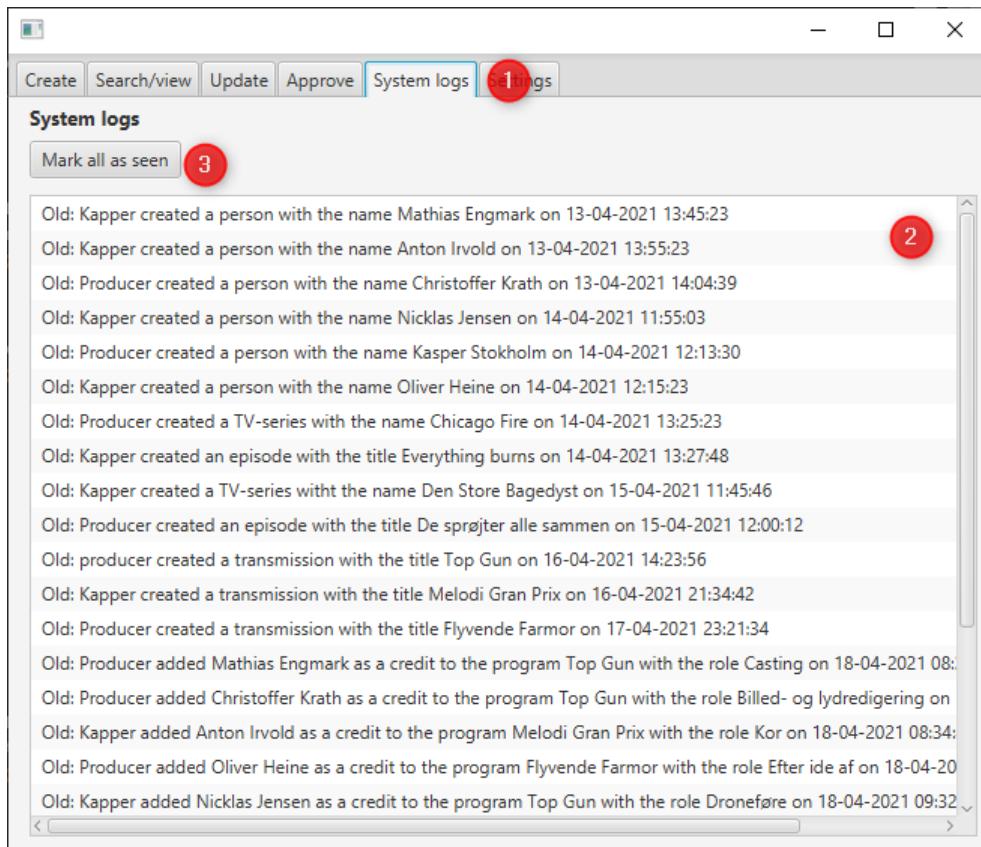
1. Under fanen "Approve".
2. Angiv programtype.
- *Hvis den angivne programtype IKKE er "TWSerie", spring punkt 3 og 4 over.*
3. Angiv TWSerien.
4. Angiv sæsonnummer.
5. Marker program.
6. Marker kreditering.
7. Tryk "Approve selected".



Figur 24 Godkende program med kreditering.

Systemadministratoren har en systemlog, hvor alle hændelser bliver noteret.
Denne er bruges således: - Se Figur 25.

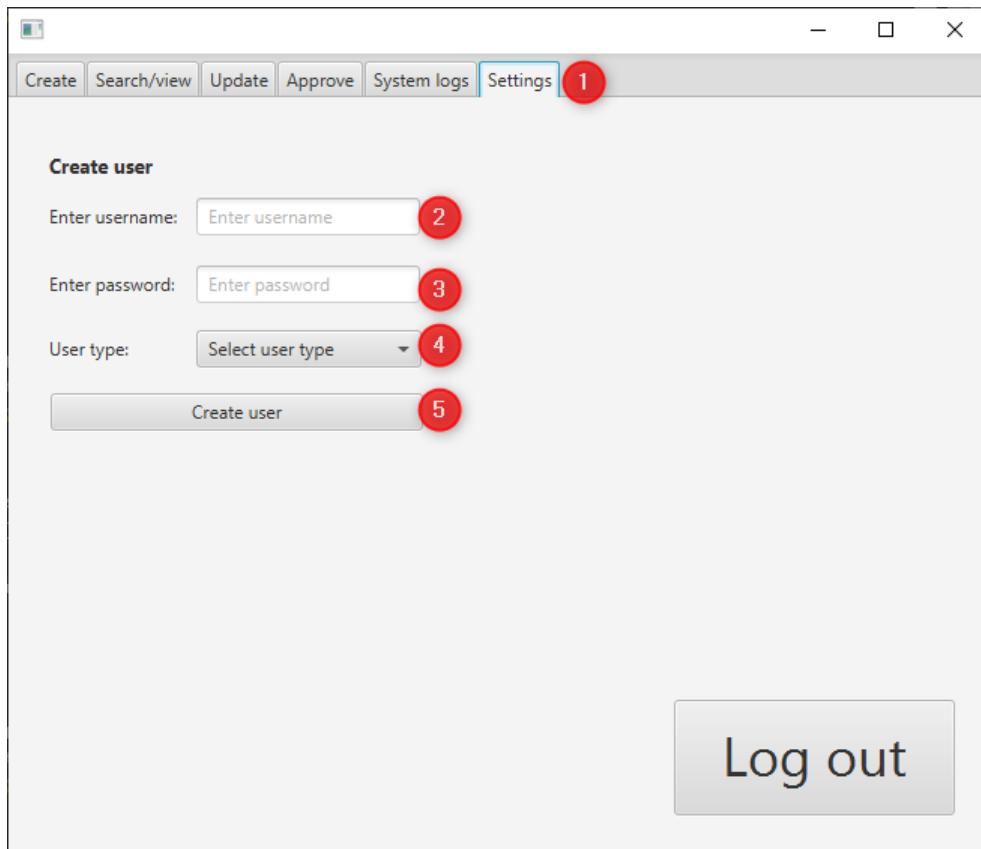
1. Under fanen "System logs".
2. En liste af hændelser, hvor nyeste er placeret nederst.
3. Marker alle som læst.



Figur 25 Godkende program med kreditering.

Som systemadministrator kan man også lave nye producere samt systemadministratore. Dette gøres således: - Se Figur 26.

1. Under fanen "Settings".
2. Indtast brugernavn.
3. Indtast adgangskode.
4. Angiv brugertype (Producer/Systemadministrator)
5. Opret bruger.



Figur 26 Oprettelse af ny producer/systemadministrator.

G Logbog

Gruppens projektlog kan findes på følgene link:

<https://github.com/mathias852/2.Semesterprojekt/wiki>

H Designklassediagram

Bilaget kan findes i mappen "Bilag" i .zip-filen

I Oversigt over kildekode

Kildekoden kan findes ved at følge følgende link:

<https://github.com/mathias852/2.Semesterprojekt/tree/main>

J Udfyldt rapportKontrolskema

Projektrapport

Projektrapporten skal have en opbygning som vist her til højre.

Projektrapporten forventes at have et omfang på 40-60 fysiske A4-sider, heraf må afsnit 8 Procesevaluering have et omfang på 5-10 A4-sider. Der er ingen begrænsning på antal sider i Bilag.

Layout, tekststørrelse, figurstørrelse, balance mellem tekst og figurer etc. vælges, så der opnås en læselig og effektiv formidling.

Projektrapporten skal overholde retningslinjer og krav i "Rapportkontolskema" (se side 2).

Rapporten skal følge retningslinjerne i

[Larsen \(2018\): Kapitel 12 Projektrapportens indhold \(side 213-246\)](#)

[Dahl mfl \(2016\): Kap. 8 uddrag vedr. evaluering](#)

[Larsen \(2018\): Kapitel 13 Professionel og klar formidling ... \(side 247-258\)](#)

i	Forside
ii	Titelblad
iii	Resumé
iv	Forord
v	Indholdsfortegnelse
vi	Læsevejledning
vii	Redaktionelt
1	Indledning
2	Faglige vidensgrundlag
3	Metoder og planlægning
4	Hovedtekst
5	Diskussion
6	Konklusion
7	Perspektivering
8	Procesevaluering
9	Referenceliste

Bilag

- A. Oversigt over kildekode
- B. Brugervejledning
- C. Samarbejdsaftale
- D. Vejlederaftale
- E. Projektlog
- F. Udfyldt rapportkontolskema
- G. Inceptionsdokument
- I. (Andre Bilag)

Rapportkontrolskema

Skemaet udfyldes ved i kolonnen "Opfyldt" at markere de krav, der menes at være opfyldt med et + og de uopfyldte med -. Skemaet indgår i rapporten som Bilag F.

Kapitel	Krav	Opfyldt +/-																
Forside	Projekttitel, uddannelsesinstitution, fakultet, institut, uddannelse, semester, kursuskode, projektperiode, vejleder, projektgruppe og projektdeltagere (fornavn, efternavn, sdu-email). Må gerne have illustrationer.	+																
Titelblad	Samme oplysninger som på forsiden, samt afleveringsdato og projektdeltagernes underskrifter (Projektdeltagernes aktive deltagelse i projektforløbet anerkendes gensidigt ved projektdeltagernes underskrifter). Må ikke have illustrationer.	+																
Resumé	<ul style="list-style-type: none"> • En kort introduktion til projektet - hvad blev der arbejdet med og hvorfor. • Problemformuleringen og vigtige afgrænsninger • Metode - hvordan angreb i problemet og hvordan realiserede i løsningen (hvem, hvad, hvornår og hvorfor) • Hovedresultater og konklusioner – hvad kom der ud af arbejdet. (max 1 side)	+																
Forord	Hensigten med rapporten, målgruppe, forhistorie, anerkendelser.	+																
Indholdsfortegnelse	Samlet indholdsfortegnelse for hele projektrapporten. Højst to eller tre niveauer i indholdsfortegnelse (der kan evt. være flere i selve rapporten). Afsnit på niveau 1 og 2 skal være nummererede.	+																
Læsevejledning	Vejledning i hvordan rapporten kan læses, eksempelvis i form af hvilken rækkefølge afsnittene kan læses i, og hvordan sammenhængen er mellem de forskellige dele af rapporten, herunder mellem hovedrapport og bilag. Rapportens målgruppe.	+																
Redaktionelt	Skriveprocessen og ansvarsområder i skriveprocessen. Ansvarsområder kan fx beskrives på fx følgende form: <table border="1"> <thead> <tr> <th>Afsnit</th> <th>Ansvarlig</th> <th>Bidrag fra</th> <th>Kontrolleret af</th> </tr> </thead> <tbody> <tr> <td>Afsnit a</td> <td>Person a</td> <td>Person b</td> <td>Person a, b, c</td> </tr> <tr> <td>Afsnit b</td> <td>Person b</td> <td>Person a</td> <td>Person a, b, c</td> </tr> <tr> <td>Afsnit c</td> <td>Person c</td> <td>Person b</td> <td>Person a, b, c</td> </tr> </tbody> </table>	Afsnit	Ansvarlig	Bidrag fra	Kontrolleret af	Afsnit a	Person a	Person b	Person a, b, c	Afsnit b	Person b	Person a	Person a, b, c	Afsnit c	Person c	Person b	Person a, b, c	+
Afsnit	Ansvarlig	Bidrag fra	Kontrolleret af															
Afsnit a	Person a	Person b	Person a, b, c															
Afsnit b	Person b	Person a	Person a, b, c															
Afsnit c	Person c	Person b	Person a, b, c															
Indledning	Projektets rammer og baggrunden for projektet. Resume af udleverede case. Problemformulering og afgrænsninger. Formål og mål med projektet. Problemformulering og afgrænsninger.	+																

	<i>(indledningen må gerne includere materiale direkte fra inceptionsdokumentet, det skal blot have en tydelig reference)</i>	
Faglige vidensgrundlag	Begrebsdefinitioner, teori og fagligt vidensgrundlag	+
Metode og planlægning	Metode Benyttede metoder i projekt (hele projektet). Kombination af UP og Scrum. Fordeler og ulemper	+
	Planlægning Plan for elaborationsfasen og de enkelte iterationer. Prioritering af krav i planlægningen. Det faktiske udviklingsarbejde. Faserne, iterationerne og det faktiske arbejde i dem? Scrum: backlogs, roller, begivenheder, scrum-buts.	+
Hovedtekst (skal indeholde resultater både fra iteration #1 og fra iteration #2)	Overordnede krav Opdateret resume af overordnede krav fra inceptionsdokument inklusive overordnet brugsmønsterdiagram og oversigt over supplerende krav	+
	Detaljerede krav Detaljeret brugsmønsterdiagram (hvis relevant). Detaljerede brugsmønsterbeskrivelser. Detaljerede beskrivelser af supplerende krav, fx organiseret efter FURPS+.	+
	Analyse Overvejelser, beslutninger og resultater vedr. analysemodellen, inklusive både den statiske og den dynamiske side af analysemodel.	+
	Design Overvejelser, beslutninger og resultater vedr. Softwarearkitektur og detaljeret design, herunder design af persistens.	+
	Databasedesign Overvejelser, beslutninger og resultater vedr. tabeldesign og SQL-forespørgsler	+
	Implementering Overvejelser, beslutninger og resultater vedr. konvertering fra design til kode illustreret gennem udvalgte centrale eksempler, samt andre vigtige implementeringsbeslutninger. Implementering af database.	+
	Test udførte test samt resultatet af dem.	+
Diskussion	Hvad er der opnået og hvad er der ikke opnået i projektet i forhold til det forventede som beskrevet i indledningen. Hvad er styrkerne og svaghederne ved resultaterne. Kunne I have opnået bedre resultater?	+

Konklusion	Opsumming af resultaterne og diskussionen af dem. Svar på problemformuleringen.	+
Perspektivering	Er den fundne løsning brugbar i anden sammenhæng? Hvad bidrager løsningen og den opnåede viden til. Fremitdigt arbejde (næste skridt i projektet, hvis I havde mere tid).	+
Procesevaluering	Processen og gruppens refleksion over processen: Læringsprocessen, teamroller, samarbejdet internt i gruppen og med vejleder, projektarbejdsformen, arbejdsformer, metoder, skriveprocessen, den tidsmæssige styring af projektet, ledelse af projektet, arbejdsfordeling i projektet m.m. Hvordan ville I gøre arbejdet an, hvis I skulle starte forfra?	+
Referenceliste	Litteratur angivet på en anerkendt form. (Alle former for litteratur som bøger, artikler og hjemmesider) Kildehenvisninger i teksten. Materiale som gruppen ikke selv har fremstillet i dette projekt skal være angivet med kilde! Alle kildehenvisninger i teksten skal være anført på samme måde. Kildeangivelser på figurer, grafer etc. som projektgruppen ikke selv har frembragt.	+
Bilag	A. Oversigt over kildekode B. Brugervejledning C. Samarbejdsaftale D. Vejlederaftale E. Projektlog F. Udfyldt rapportkontrolskema G. Inceptionsdokument I. (Andre Bilag)	+

Rapporttekniske elementer		Opfyldt +/-
Layout	Er der anvendt samme layout i alle kapitler. Er layout overskueligt/harmonisk.	+
Sprog	Formidler rapporten projektet faglig og sagligt. Er sproget neutralt, aktivt, upersonligt, konkret, præcist, kortfattet og korrekt. (Procesevalueringen må benytte personligt sprog)	+
Sidenummerering	Er der korrekt og konsistent sidenummerering i rapporten.	+
Figurer/diagrammer	Er alle figurer konsekvent nummererede. Er der figurtitel og figurtekst til alle figurer. Er figurtitler og figurtekster dækkende og afklarende. Er figurerne tydelige og læsbare. Er figurerne informationsgivende og i den rette sammenhæng.	+
Tabeller	Er alle tabeller konsekvent nummererede. Er der en forklarende tabeltekst til alle tabeller. Er alle søjler og rækker forsynet med parametre. Er der enheder på alle relevante rækker og søjler.	+

Sporbarhed af begreber	Er der en konsekvent brug af samme betegnelse for et givet begreb igennem rapporten.	+
-------------------------------	--	---

K Inceptionsdokument

*Se næste side. Bemærk, sidetallene passer ikke for denne rapport på de følgende sider.
Dette er overskrevet af Inceptionsdokumentets side-tal.*



University of
Southern Denmark

Inceptionsdokument

UNIVERSITY OF SOUTHERN DENMARK

FACULTY OF ENGINEERING

MAERSK MC-KINNEY MOLLER INSTITUTE

2. SEMESTERPROJEKT SOFTWARETEKNOLOGI

KURSUS KODE - ST2-PRO

PROJEKT PERIODE: FORÅR 21: 02. FEBRUAR 2021 - 28. MAJ 2021

Gruppe ST 01

Mathias Jeppesen Engmark
Oliver Heine
Anton Valdemar Dahlin Irvold
Nicklas Bruun Jensen
Christoffer Schurmann Krath
Kasper Stokholm

maeng20@student.sdu.dk
olhei20@student.sdu.dk
anirv20@student.sdu.dk
nickj20@student.sdu.dk
chkra19@student.sdu.dk
kasto16@student.sdu.dk

Vejleder: Henrik Lange

SDU

II Titelblad

UNIVERSITY OF SOUTHERN DENMARK

FACULTY OF ENGINEERING

MAERSK MC-KINNEY MOLLER INSTITUTE

2. SEMESTER PROJEKT SOFTWARE

KURSUS KODE - SE2-PRO

PROJEKT PERIODE: 02. FEBRUAR 2021 - 28. MAJ 2021

Forfattere

Mathias J. Engmark

Mathias Engmark

Oliver Heine

Oliver Heine

Anton V. D. Irvold

Anton Irvold

Nicklas B. Jensen

Nicklas Jensen

Christoffer S. Krath

Christoffer Krath

Kasper Stokholm

Kasper Stokholm

III Indholdsfortegnelse

1 Indledning	1
1.1 Projektets rammer og baggrunden for projektet	1
1.2 Formål med- og mål for inceptionsfasen	1
1.3 Problemanalysen	1
1.4 Problemformulering og afgrænsninger	2
1.4.1 Idéer til Credits Management-systemet	2
2 Fagligt vidensgrundlag	4
2.1 Begrebsdefintioner	4
2.2 Teori og viden	4
2.2.1 Unified Process	5
2.2.2 Scrum	6
2.2.3 Unified Modelling Language	6
2.3 Allerede eksisterende problemløsninger	6
3 Overordnet kravspecifikation	7
3.1 Aktørliste	7
3.2 Brugsmönsterliste	9
3.3 Supplerende krav	10
3.4 Brugsmönsterdiagram for Credits Management System	11
3.5 Detaljeret brugsmönsterbeskrivelse	12
4 Kritiske risici	16
5 Prioritering	18
6 Metode i Elaborationsfasen	19
6.1 Kombination af Unified Process og Scrum	19
7 Ressourcer	20
7.1 Arbejdstid	20
8 Konklusion	21
Bibliografi	22
9 Procesdokumenter	23
9.1 Gruppekontrakt	23
9.2 Vejderkontrakt	26
9.3 Belbin-Teamprofil	27
Appendices	30

A	Checkliste for inceptionsdokument	31
B	Logbog	33
C	Referat fra reviewmøde	33

KAPITEL 1

Indledning

Følgende afsnit vil belyse baggrunden for projektet og formålet med rapporten. Ydermere vil der blive udarbejdet en problemanalyse, hvilken danner grundlag for problemformuleringen og projektets afgrænsning.

1.1 Projektets rammer og baggrunden for projektet

Projektet er udarbejdet i forbindelse med semesterprojektet på andet semester ved ingeniøruddannelsen Softwareteknologi. Semesterprojektet er bygget op omkring en udleveret case fra virksomheden TV2.

1.2 Formål med- og mål for inceptionsfasen

Inceptionsfasen i dette projekt, har til formål at opstille projektets fremtidig rammer, samt at analysere og vurdere kritiske krav til systemet og produktudviklingen, som afgører fortsættelsen af produktudviklingen. Ydermere opstilles en overordnet brugsmønstermodel, som har til formål at definere produktudviklingens afgrænsning og krav.

Det er essentielt for yderligere udvikling, at inceptionsfasen vurderes tilstrækkelig succesfuld, før projektet kan fortsætte til elaborationsfasen.

1.3 Problemanalysen

Med basis i TV2's udleverede case, skal der laves et Credits Management System, som er tilgængeligt for seere af TV2 samt diverse administrerende roller indenfor TV2 og Nordisk film. Systemet er et eksternt system, hvilket håndterer visning, oprettelse og registrering af kreditering. Til håndtering af data bruges en database lavet i PostgreSQL, hvor en applikation, skrevet i Java, håndterer brugerfladen for både normale brugere og 'super-brugere' af systemet.

TV2 ønsker et system, der kan fjerne tv-tiden brugt på rulletekster og kreditering, så

denne tid i stedet kan bruges på TV2's egne marketing projekter. Desuden ønsker TV2 at automatisere de, i dag, meget manuelle arbejdsprocessor, og desuden udlicitere arbejdet til producenterne selv. Problemet indebærer også mere automatiseret eksport af den data, som krediteringerne danner, således at de krediterede personer kan blive betalt passende. Det endelige mål for TV2 er, at TV2 udelukkende skalstå for manuel kvalitetssikring af data og vedligeholdelse af systemet.

1.4 Problemformulering og afgrænsninger

Problemformuleringen fungerer som projektets afgrænsnings for den udleverede case af TV2. Ydermere er der opsat underspørgsmål, hvilket er med til at afgrænse projektets retning. Problemformuleringen er som følgende:

Hvordan kan man opstille et Credits Management System, som gør det muligt for TV2 at afskaffe krediteringstiden på programmer?

Nedenstående er en afgrænsning af problemformuleringen:

- Hvordan kan man lave et softwaresystem, som håndterer forskellige brugers roller og muligheder i et Credits Management System?
- Hvordan kan en database til et Credits Management System opsættes?
- Hvordan kan man lave en brugergrænseflade til et Credits Management System?
- Hvordan kan man lave et notifikationssystem, der underretter de relevante brugere om ændringer i Credits Management-systemet?

1.4.1 Idéer til Credits Management-systemet

Nedenstående er en liste over en tidlig brainstorm af idéer til Credits Management-systemet. Idéerne er ikke rangeret efter vigtighed men blot en oversigt over, hvordan gruppen synes, at en løsning af TV2's case *kunne* se ud endeligt. Disse idéer har dannet grundlag for ovenstående problemformulering.

- Database over al krediteringsinformation.
- Oprettelse af en backup af databasen, eksempelvis hver 4. time.
- GUI til styring af database med mulighed for at trække data ud om specifikke personer, roller i produktionen, film og programmer.
- Eksportering af data til XML-fil og CSV-fil.
- Unik identifikation af personer i en produktion i systemet fx gennem UUID.

- En form for access control – forskel på administrator og bruger. Forskel på funktionalitet i programmet alt efter ”rolle” i systemet:
 - Gennem en fælles klient kan man som ”super-user” logge ind og tilgå systemet med flere funktioner. Her kan der eksempelvis findes muligheder for administration af systemet samt et notifikationssystem, når der laves ændringer. Notifikationssystemet skal opdateres med en ”refresh” knap og skal kun opdateres for relevante roller.
 - Mulige roller i systemet: bruger, producer, system maintainer, system admin.
- Eksportering af program som .exe
- QR-kode eller Google Chrome extensions til brug af fysiske/webbaserede applikationer som TV2Play, YouSee Play osv.

KAPITEL 2

Fagligt vidensgrundlag

Kommende afsnit dækker over det faglige vidensgrundlag, herunder begrebsdefintion, teori samt de metoder, som rapporten vil læne sig op ad.

2.1 Begrebsdefintioner

Brugte begreber igennem inceptionsdokumentet er formelt definitioner i tabel 2.1. Yderligere defination af begreberne, vil ikke finde sted i rapporten.

Credits Management System	Dette er systemet som udarbejdes og dækker over håndtering af credits til film, TV-serier samt andet TV-indhold
Electronic Program Guide	Forkortes EPG. Dette er systemet som giver publications til vores Credits Management System.
Publications	Givet af EPG. Publications indeholder data om diverse TV-indhold.

Tabel 2.1 Begrebsdefinitioner.

2.2 Teori og viden

Gruppen studerer softwareteknologi på 2. semester, og har gennemført følgende kurser på første semester:

- Computersystemer.
- Objektorienteret Programmering.
- Problemorienteret Projektarbejde.
- Semesterprojekt: Udvikling af Softwareprogrammer.
- Statistisk Dataanalyse.

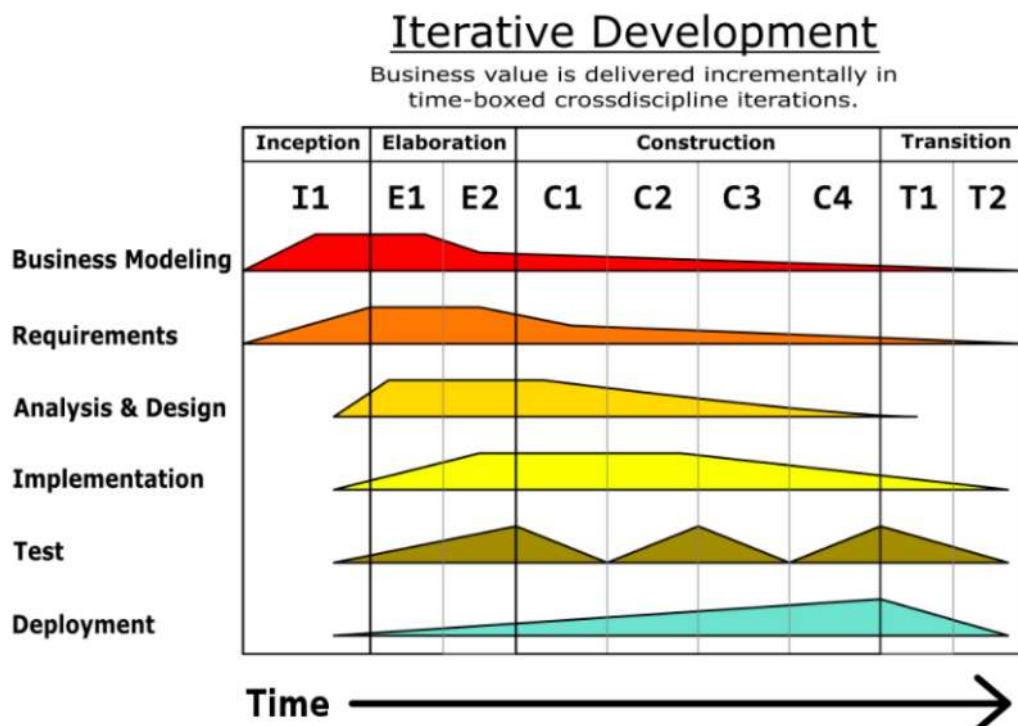
Gruppen vil i løbet af 2. semester tilegne sig yderligere viden igennem følgende kurser:

- Grundlæggende Software Engineering.
- Data Management.
- Videregående Objektorienteret Programmering.
- Cyber-Physical Systems.

Det forventes at læseren har tilsvarende eller bedre viden for at kunne forstå rapporten. Projektet vil gøre brug af en kombination af Unified Process (UP) og Scrum, hvilket vil blive uddybet i afsnit 6.1 Metode i Elaborationsfasen.

2.2.1 Unified Process

Unified Process(UP) er en agil arbejds metode som strækker sig over fire faser; *Inceptionsfasen*, *Elaborationsfasen*, *Konstruktionsfasen* & *Transitionsfasen*. Det vurderes, at læseren har forståelse for arbejds metoden, men grundlæggende bliver et projekt udført i overensstemmelse mellem de fire faser, hvor hver fase har sine fokusområder, se figur 2.1. Følgende rapport vil kun gøre brug af faserne *inceptionsfasen* & *elaborationsfasen*.



Figur 2.1 Oversigt over UP-faserne [1]

Inceptionsfasen indebærer forståelse for, hvad projektet omhandler og hvad der skal udvikles. Opstiller krav til systemet i form af brugsmønstre. Derudover tages der stilling

til projektets risici samt planen for projektet[2].

Elaborationsfasen indebærer et design af systemets arkitektur samt design af førhen beskrevne brugsmønstre. Derudover udvikles en designmodel og dertilhørende test-cases. ydermere udvikles en fungerende prototype. Endeligt bliver projektplanen omstruktureret qua ny viden[3].

2.2.2 Scrum

Scrum er en agil softwareudviklingsmetode, som benytter sig af fleksible tidsplaner, deadlines samt fokuseret arbejde på specifikke mål.

Projektets planlægning ved elaborationsfasen vil blive struktureret ved hjælp Scrum. Kombinationen mellem Scrum og kombineres bliver beskrevet nærmere i afsnit 6.1 Metode i elaborationsfasen.

2.2.3 Unified Modelling Language

Unified Modelling Language (UML) er et velkendt modelleringssporg for diagrammer til udvikling og dokumentation af software. [4] Rapporten vil fokusere på et brugsmønster-diagram for at opstille krav til systemet ud fra specifikke situationer.

2.3 Allerede eksisterende problemløsninger

Den nuværende løsning til problemet består i den kendte 'TV-guide', som efterhånden alle TV har - i denne guide kan en komprimeret udgave af credits til TV-indslaget findes. Heri findes der oftest kun information i form af instruktøren af filmen samt en liste over skuespillere.

TV2s nuværende arbejdssproces foregår ved at producenten sender en word eller pdf-fil til TV2, hvorefter TV2 manuelt indtaster denne data i deres system. Disse filer bliver delt over file-sharing programmer.

KAPITEL 3

Overordnet kravspecifikation

Følgende afsnit vil udarbejde projektets overordnede kravspecifikation igennem en grundlæggende brugsmönstermodel, hvilken danner grundlag for opgavens videre udvikling samt projektets supplerende krav.

3.1 Aktørliste

For at få en dybere forståelse af, hvem som har til hensigt at interagere med systemet, opstilles en aktørliste, som beskriver hvem aktørerne er, samt deres handling og intention med systemet. Det skaber ydermere en forståelse af, hvordan systemet skal fungere, og hvad det skal indeholde. Aktørerne i tabel 3.1 er de primære aktører, som ønsker at interagerer med systemet, hvor tabel 3.2 beskriver de aktører, som hjælper med udførsel af bestemte interaktioner fra de primære aktører.

<i>Aktørliste</i>	
Navn	Beskrivelse
Gæst	Privatperson, der gerne vil bruge systemet til at finde kreditering for en produktion. Denne privatperson behøver ikke at logge ind for at bruge systemet. En gæst skal kunne: <ul style="list-style-type: none">• Søge efter kreditering• Logge ind• Oprette en bruger

KAPITEL 3. OVERORDNET KRAVSPECIFIKATION

Bruger	<p>Privatperson der har oprettet en bruger på systemet. Systemet vil blive brugt af bruger på samme måde som af en gæst, men har andre funktioner tilgængelige til personlig-gørelse af oplevelsen.</p> <p>En bruger skal kunne:</p> <ul style="list-style-type: none"> • Søge efter kreditering • Gemme en kreditering som favorit • Abonnere på en krediteret person • Tilgå favorit kreditering • Logge ud • Redigere egne kontoooplysninger
Producer	<p>Person, som kan oprette og ændre programmer, samt kreditering af produktioner de selv ejer.</p> <p>En producer skal kunne:</p> <ul style="list-style-type: none"> • Søge efter kreditering • Oprette et program med kreditering • Opdatere kreditering på program de ejer • Logge ud
Systemadministrator/System maintainer	<p>Person som har mulighed for at oprette og opdatere samt fjerne kreditering og produktioner. Ydermere kan systemadministrator også oprette, opdatere og fjerne brugere fra systemet.</p> <p>En systemadministrator skal kunne:</p> <ul style="list-style-type: none"> • Ændre bruger • Tildle specielle adgangsrettigheder for anden bruger • Ændre adgangsrettigheder for anden bruger • Oprette et program med kreditering • Slette kreditering • Opdatere program • Slette program • Eksportere kreditering • Godkende kreditering

Tabel 3.1 Liste over primære aktører

<i>Sekundær Aktørliste</i>	
Navn	Beskrivelse
Electronic Program Guide (EPG) (Sekundær)	Det er meningen, at systemet skal kunne importere data fra EPG.

Tabel 3.2 Liste over sekundære aktører.

3.2 Brugsmønsterliste

Aktørlisten i tabel 3.1 viser hvad de forskellige aktører skal kunne foretage sig i systemet, men for at uddybe hver interaktion oprettes en brugsmønsterliste. Brugsmønsterlisten, tabel 3.3, har til formål at beskrive og forklare hver enkel interaktion i detalje, hvilket skaber en bedre forståelse over de forskellige interaktioner, systemet skal kunne håndtere. Aktør-kolonnen viser kun de aktører, hvis primære interaktion med systemet er det gældende brugsmønster.

<i>Brugsmønsterliste</i>			
ID	Navn	Beskrivelse	Aktør
B01	Søg efter kreditering	En gæst skal søge efter et program eller en kreditering (eksempelvis skuespiller, producent eller fotograf), få en liste af søgeresultater der matcher kriterierne, og vælge det ønskede resultat.	Gæst(p), Bruger(p)
B02	Se egne kontooplysninger	En bruger skal kunne se sine indtastede kontooplysninger.	Bruger(p)
B03	Rediger egne kontooplysninger	En bruger skal kunne redigere i sine kontooplysninger.	Bruger(p)
B04	Logge ind	En gæst skal med korrekte legitimationsoplysninger kunne logge ind på systemet og tilgå deres personlige konto.	Gæst(P)
B05	Oprette en bruger	En gæst skal kunne oprette en bruger.	Gæst(p)
B06	Logge ud	En bruger som er logget ind med korrekte legitimationsoplysninger, skal kunne logge ud igen.	Bruger(p)
B07	Gemme en kreditering som favorit	En bruger på systemet skal kunne gemme sine favorit krediteringer som en ‘favorit’ for at kunne tilgå dem på et senere tidspunkt.	Bruger(p)
B08	Tilgå favorit kreditering	Brugeren skal kunne tilgå de krediteringer som vedkommende har favoriseret.	Bruger(p)
B09	Abonnere på en krediteret person	En bruger som er logget ind, skal kunne abonnere på en krediteret person. Dette giver dem notifikationer når personen bliver krediteret for noget nyt.	Bruger(p)
B10	Oprette et program med kreditering (producer)	En producent skal oprette et program i systemet. Når programmet oprettes, skal der tilføjes relevant information om programmet, og tilføjes relevante krediteringer.	Producer(p), EPG(s)
B11	Opdatere kreditering på program	En producent skal kunne opdatere krediteringsoplysningerne for et program de ejer.	Producer(p)

B12	Eksportere kreditering	Systemadministrator skal kunne eksportere krediteringer til manuel brug af dataanalyse.	Systemadministrator (p)
B13	Godkende en nyoprettet kreditering	Systemadministratoren skal have mulighed for at godkende krediteringer i programmer. Systemadministrator (p)	
B14	Ændre bruger	Systemadministrator skal kunne ændre eksisterende brugeres oplysninger.	Systemadministrator(p)
B15	Ændre adgangsrettigheder for anden bruger	Systemadministratoren skal kunne ændre adgangsrettigheder for oprettede brugere.	Systemadministrator(p)
B16	Slette kreditering	Systemadministratoren skal kunne slette eksisterende krediteringer	Systemadministrator(p)
B17	Opdatere program	Systemadministratoren skal kunne opdatere programmer og deres dertilhørende krediteringer	Systemadministrator(p), EPG(s)
B18	Slette program	Systemadministratoren skal kunne slette eksisterende programmer.	Systemadministrator(p)

Tabel 3.3 Brugsmønsterliste

3.3 Supplerende krav

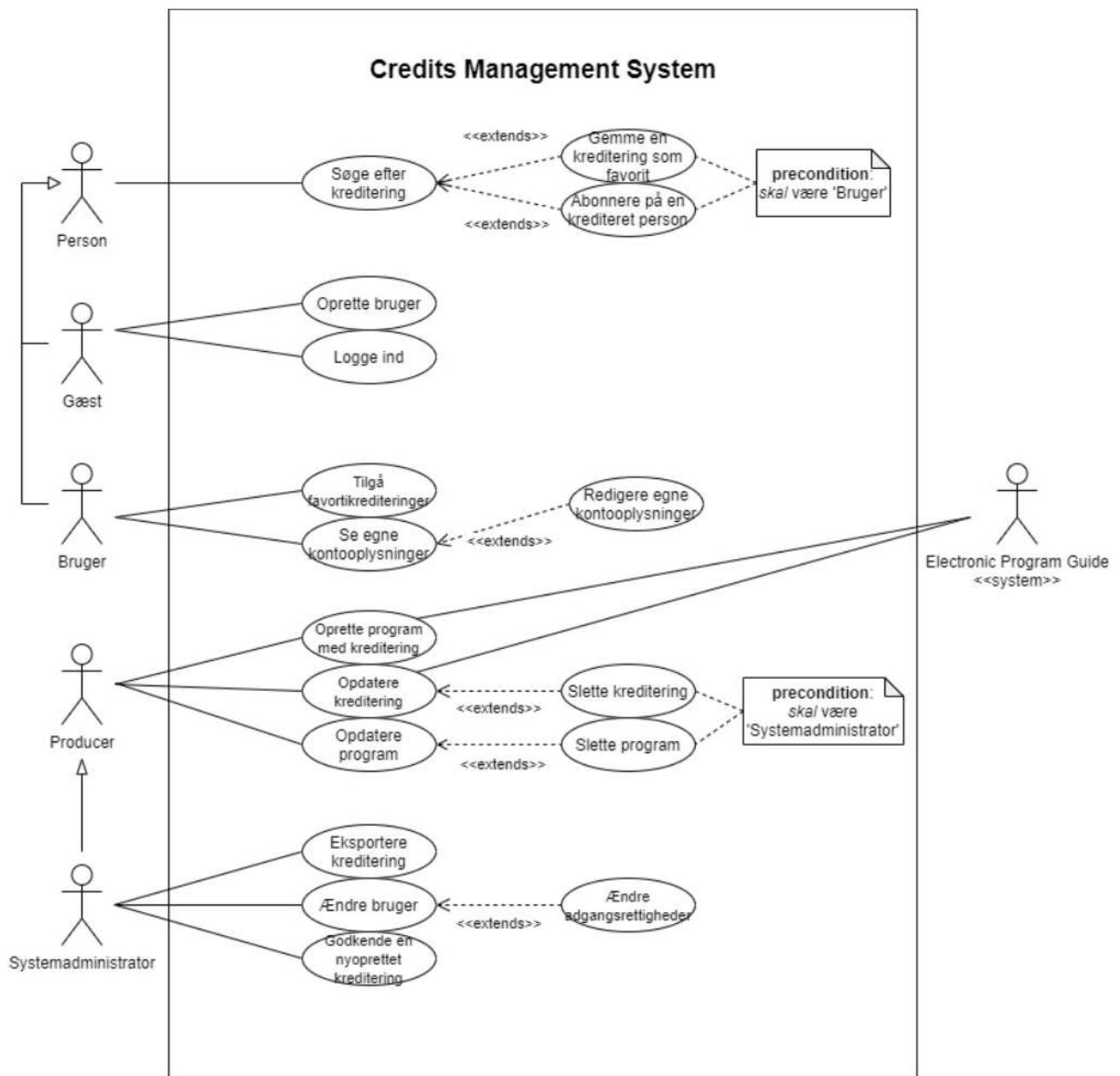
Tabel 3.4 viser de supplerende krav til systemet. Disse krav er ikke-funktionelle krav, hvilke definere nogle vilkår til implementeringen. Ydermere fungerer de som begrænsninger på systemet.

Supplerende krav			
ID	Type	Navn	Beskrivelse
S01	Data integrity	Dataeksport	Krediteringen, som eksporteres, skal være utsynlig og kvalitetssikret.
S02	Manageability	Opdatering af krediteringer	Når en kreditering opdateres, skal denne kreditering stadigvæk vises og systemet skal virke.
S03	Security	Adgangsrettigheder	Kun aktører med rollen "Producer" eller "Systemadministrator" har adgang til information om krediteret personer, som de krediterede personer ikke har lyst til at oplyse til andre.
S04	Usability	Brugervenlig-Interface	Credit management systemets interface, skal være brugervenligt og intuitivt at forstå/bruge.

Tabel 3.4 Supplerende krav

3.4 Brugsmønsterdiagram for Credits Management System

Nedenstående ses et brugsmønsterdiagram for Credits Management-systemet som er udviklet på baggrund af identificerede aktører samt deres brugsmønstre. Brugsmønsterdiagrammet indeholder aktørenes primære brugsmønstre, og vil derfor ikke være en komplet udgave af tabel 3.1.



Figur 3.1 Brugsmønsterdiagram over Credits Management System

3.5 Detaljeret brugsmønsterbeskrivelse

På baggrund af den nye forståelse for systemet og dets funktioner, er der udvalgt tre brugsmønstre, som vurderes essentielle for systemet, baseret på casen fra TV2. De udvalgte brugsmønstre beskrives med en større detaljegrad, for at give en dybdegående forståelse for de pågældende brugsmønstre. Det er samtidig med til at give en forståelse for, hvilke krav som eventuelt skal være opfyldt før og efter brugsmønstrets hændelsesforløbet, samt hvilke alternative hændelser, som kan opstå. De detaljeret brugsmønstre kan være med til at hjælpe fremtidig udvikling.

Brugsmønster: Søge efter kreditering	
ID:	B01
Primære aktører:	Bruger, Gæst
Sekundære aktører:	Ingen
Kort beskrivelse:	En bruger eller gæst - omtales aktør - skal søge efter et program eller en kreditering (eksempelvis skuespiller, producer eller fotograf), få en liste af søgeresultater der matcher kriterierne, og vælge det ønskede resultat.
Prækonditioner (Pre-conditions):	<ul style="list-style-type: none"> • Ingen
Hovedhændelses-forløb:	<ol style="list-style-type: none"> 1. Brugsmønsteret starter når aktøren vil søge på et program eller krediteret person. 2. Aktøren kan vælge ønsket søgekategori <ol style="list-style-type: none"> 2.1. Følgende søgekategori består af: <ol style="list-style-type: none"> 2.1.1. Serie, film, program 2.1.2. Krediteringsrolle 3. Aktøren søger på søgningskriteriet 4. Systemet lavet et kald til databasen med pågældende søgekriterie 5. Systemet modtager et svar fra databasen 6. Systemet viser relevante oplysninger efter det ønskede kriterie til aktøren
Postkonditioner (Post-conditions):	<ul style="list-style-type: none"> • Søgekriteriet gemmes til brug i databehandling • Aktøren får fremvist relevante oplysninger baseret på indtastede søgekriterier

Alternative hændelsesforløb (Alternative flow):	<ul style="list-style-type: none"> • Aktøren kan afslutte brugsmønsteret fra punkt 1-3 • Kreditering med søgekriteriet findes ikke
--	--

Tabel 3.5 Søg kreditering

Brugsmønster: Eksportering af kreditering	
ID:	B12
Primære aktører:	Systemadministrator, Producer
Sekundære aktører:	Ingen
Kort beskrivelse:	Systemadministrator og producer skal kunne eksportere krediteringer til manuel brug af dataanalyse.
Prækonditioner (Pre-conditions):	<ul style="list-style-type: none"> • Aktøren skal være logget ind • Der skal eksistere krediteringer i systemet
Hovedhændelses-forløb:	<ol style="list-style-type: none"> 1. Brugsmønsteret starter når aktøren vil eksportere en eller flere krediteringer 2. Aktøren indtaster ønskede søgekriterie 3. Aktøren søger efter ønskede krediteringer 4. Systemet sender søgningen til databasen 5. Systemet modtager data fra databasen 6. Systemet viser data til aktøren 7. Aktøren markerer viste kreditering 8. Aktøren eksporterer markerede krediteringer <ol style="list-style-type: none"> 8.1. Aktøren vælger ønsket filformat og filplacering 9. Systemet eksporterer viste markering i ønsket filformat til ønsket filplacering
Postkonditioner (Post-conditions):	<ul style="list-style-type: none"> • Systemet opretter og gemmer aktørens handlinger i systemet

Alternative hændelsesforløb (Alternative flow):	<ul style="list-style-type: none"> • Aktøren kan afslutte brugsmønsteret fra punkt 1-8 • Kreditering med søgekriterie findes ikke. • Hvis filformat og placering ikke bliver angivet, sættes dette til en standard værdi • Hvis aktøren ikke har markeret viste krediteringer (punkt 7) kan brugsmønsteret (punkt 8-9) ikke fortsætte.
--	--

Tabel 3.6 Eksportering af kreditering

Brugsmønster: Oprette kreditering	
ID:	B10
Primære aktører:	Producer, Systemadministrator
Sekundære aktører:	EPG-Systemet
Kort beskrivelse:	En producer eller systemadministrator skal oprette et program i systemet. Når programmet oprettes, skal der tilføjes relevant information om programmet, og tilføjes relevante krediteringer. Derefter sendes en forespørgsel til systemadministrator, som skal godkende krediteringen.
Prækonditioner (Pre-conditions):	<ul style="list-style-type: none"> • Produceren skal være logget ind
Hovedhændelses-forløb:	<ol style="list-style-type: none"> 1. Brugsmønstret starter, når aktøren skal oprette en kreditering 2. Aktøren trykket på knappen “Opret ny kreditering” 3. Systemet kræver aktørens log-in igen (include B02 (Login)) 4. Systemet fører aktøren til et nyt vindue, hvor krediteringer kan oprettes 5. Aktøren udfylder de relevante felter og krediteringer 6. Aktøren trykket på knappen “Opret kreditering” 7. Systemet sender de oprettede krediteringer til databasen 8. Systemet viser aktøren den oprettede kreditering <ol style="list-style-type: none"> 8.1. Vises med en status som “afventer” 9. Systemet gemmer krediteringen og sender en forespørgsel til systemadministrator 10. Include B11 (Godkendelse af nyoprettet krediteringer)

Postkonditioner (Post-conditions):	<ul style="list-style-type: none"> • Den oprettede kreditering skal godkendes af systemadministratoren
Alternative hændelsesforløb (Alternative flow):	<ul style="list-style-type: none"> • Aktøren kan afslutte brugsmønsteret overalt fra punkt 2-6 • Hvis aktørens login (punkt 3) ikke er korrekt, afbrydes brugsmønster • Hvis forbindelsen til databasen afbrydes, får aktøren besked, og brugsmønsteret afbrydes • Aktøren vil gerne stoppe oprettelsen midlertidigt (Punkt 5). <ul style="list-style-type: none"> – Aktøren trykker på knappen “gem” under processen i punkt 5 – Ved tryk på knappen “gem” gemmer systemet indskrevet data fra felterne

Tabel 3.7 Oprettelse af kreditering.

KAPITEL 4

Kritiske risici

Et projekt vil naturligvisstå overfor udfordringer, som kan true projektets udfald. Selvom det ikke altid kan undgås, er det muligt at mindske påvirkning ved forberedelse. På baggrund deraf, er tabel 4.1 udarbejdet, som hjælp til risikostyring, hvilken har til formål at forberede projektgruppen på tænkelige udfordringer, som kan true projektets kvalitet eller tidsplan. Tabellen indeholder en beskrivelse af risikoen, samt en plan for, hvordan truslen kan mitigeres.

ID	Navn	Påvirker	Beskrivelse	Mitigering
R01	Sygdom eller fravær	Projekt	Et eller flere gruppemedlemmer, som har skelsættende viden om en del af projektet/produktet, ligger sig syg eller er af anden grund fravaerende.	Denne risiko kan mitigeres ved, at der hele tiden deles viden imellem gruppen, så alle har samme udgangspunkt for arbejdet.
R02	Kode går tabt	Projekt & produkt	Hvis af en grund der går kode tabt eksempelvis ved fejl i brug af Git.	Alle i gruppen sætter sig på forhånd ind i Git, så der ikke er tvivl om, hvordan det bruges. Kan også undgås ved at tage backups jævnligt.
R03	Tidsplan skubbet grundet undervisningsforsinkelse	Projekt	Projektarbejdet er baseret på undervisningen. Derfor vil projektets tidsplan også skubbes, hvis undervisningen skubbes.	Der gøres plads i tidsplanen i slutningen af projektet til at indhente tidsplanen.
R04	Teknologi-begrænsning	Produkt	Hvis teknologien, der skal bruges i projektet, ikke kan håndtere opgaver og krav opstillet af gruppen.	Finde alternative muligheder for løsninger til opgaverne og eventuelt ændre i krav i samarbejde med kunden.
R05	Ændring af krav fra kunde	Projekt	Hvis kundens krav og behov ændrer sig, vil gruppen blive nødsaget til at ændre krav og analysearbejdet.	Der skubbes i tidsplanen, og der arbejdes ekstra eller gøres brug af pusterummet til sidst i tidsplanen.

Tabel 4.1 Beskrivelse af kritiske risici.

På baggrund af tabel 4.1 beregnes risikostørrelse for hver risiko, hvilket har til formål, at give en fornemmelse af truslen. Risikostørrelsen vil blive udregnet på baggrund af

KAPITEL 4. KRITISKE RISICI

den multiplicerede værdi mellem sandsynligheden og alvorligheden. Dette er vist i tabel 4.2. Værdien for sandsynligheden samt alvorligheden vil blive sat af selve projektgruppen mellem 1-5, hvor 1 er lavest og 5 er højest. Heraf afsløres de største risici, som gruppen vil være opmærksomme på igennem arbejdsprocessen.

ID	Sandsynlighed	Alvorlig	Risikostørrelse	Note
R01	2	3	6	Normalt ville sandsynligheden være sat til 1. Grundet COVID, sættes sandsynligheden til 2.
R02	3	3	9	Alvorligheden afhænger af størrelsen af fejlen.
R03	2	1	2	
R04	1	1	1	
R05	1	3	3	

Tabel 4.2 Kritiske risici tabel.

KAPITEL 5

Prioritering

Prioriteringerne i tabel 5.1 tager udgangspunkt i kravene fundet igennem brugsmønstrene ved kapitel. For prioritering, vil der gøres brug af MoSCoW metoden. Der er desuden prioriteret internt indefor hver kategori; jo højere oppe i listen, jo højere prioritet.

ID	Navn	Prioritet
B10	Oprette et program med kreditering (Producer)	M (1.)
B13	Godkende en nyoprettet kreditering	M (2.)
B11	Opdatere kreditering på program en specifik producer ejer	M (3.)
B17	Opdatere program	M (4.)
B04	Logge ind	M (5.)
S03	Adgangsrettigheder	M (6.)
B06	Logge ud	M (7.)
B16	Slette kreditering	M (8.)
B18	Slette program	M (9.)
B01	Søg og gennemse kreditering	M (10.)
B05	Oprette en bruger	S (11.)
S02	Opdatering af krediteringer	S (12.)
B12	Eksportere kreditering	S (14.)
S01	Dataeksport	S (15.)
B14	AEndre bruger	S (16.)
S04	Brugervenlig-Interface	S (17.)
B15	AEndre adgangsrettigheder for anden bruger	C (18.)
B07	Gemme en kreditering som favorit	C (19.)
B08	Tilgå favorit kreditering	C (20.)
B09	Abonnere på en krediteret person	W (21.)

Tabel 5.1 MoSCoW-prioritering.

KAPITEL 6

Metode i Elaborationsfasen

6.1 Kombination af Unified Process og Scrum

Følgende projekt vil gøre brug af UP, men med en kombination af Scrum. Første del af projektet vil følge den grundlæggende arbejdsmetode for inceptionsfasen, men ved påbegyndelse af elaborationsfasen vil UP blive brugt i en kombination med Scrum. Elaborationsfasen vil samtidig være den afsluttende fase for projektet, hvilket skyldes den case, som projektgruppen følger. UP vil være den generelle arbejdsmetode mens planlægningen for projektets elaborationsfase vil udføres gennem Scrum. Der vil blive arbejdet over to iterationer, også kendt som sprints ved Scrum, som følger oversigten på figur 2.1 for elaborationsfasen. Hensigten med kombinationen af UP og Scrum er, at nå milepælen for elaborationsfasen. Yderligere fordele ved kombinationen er, at gruppen bag projektet ikke har den største erfaring med projektudvikling, og har derved større kontrol over eventuelle ændringer ved planlægning af fremtidig arbejde. Det skyldes, at planlægningen ved Scrum lægger vægt på det forrige sprint, og er altså ikke bestemt på forhånd. Samtidig vil det give bedre indsigt i den agile arbejds metode, som vil styrke fremtidig arbejde.

KAPITEL 7

Ressourcer

7.1 Arbejdstid

Gruppen har registreret arbejdstider uge for uge i inceptionsfasen. Foreløbige tal ses i tabellen nedenfor. Timerne er for fælles arbejdstimer, og tager ikke højde for timer, som gruppemedlemmerne har brugt individuelt.

Uge nr.	Timer
6	7 timer
7	11 timer
8	10 timer
9	12 timer
10	12 timer
11	10 timer
I alt:	62 timer

Tabel 7.1 Arbejdstsidsindsats uge for uge og alt i alt.

Gruppen har desuden lavet et Gantt diagram som viser, hvordan ovenstående arbejdstimer ved inceptionsfasen, er blevet brugt uge for uge. Ydermere ses et grønt markeret felt med teksten *Scrum*, hvilket viser perioden, hvor Gantt ikke vil blive brugt, da Scrum ikke understøtter planlægningen i Gantt. Det skyldes overgangen til planlægningsmetoden Scrum, som blev beskrevet i kapitel 6.1



Figur 7.1 Gantt-diagram lavet for projektet.

KAPITEL 8

Konklusion

Baseret på den udleverede case fra TV2 omhandlende et Credits Management System, som skal håndtere kreditering af film og TV-programmer, har gruppen udarbejdet en problemformulering til virksomheden, som lyder: '*Hvordan kan man opstille et Credits Management System, som gør det muligt for TV2 at afskaffe krediteringstiden på programmer?*'. Problemformuleringen har en tilhørende afgrænsning, der er med til at specificere projektet.

I henhold til den udleverede case blev en overordnet brugsmønstermodel udviklet. Relevante aktører for Credits Management-systemet blev identificeret, som dannede grundlag for opstillingen af en brugsmønsterliste og udvikling af et brugsmønsterdiagram. Den overordnede brugsmønstermodel danner rammen for hvilke funktionelle krav systemet skal indeholde. Ydermere blev supplerende krav opstillet, som gav et overblik over de ikke-funktionelle krav for systemet. Begge blev senere prioriteret ved brug af MoSCoW metoden. Brugsmønstermodellen har været med til at definere og identificere projektets scope til fremtidig udvikling samt bringe en forståelse for den udleverede case. Den har også givet gruppen et bedre overblik over, hvilket produkt der ønskes udviklet.

Der blev udarbejdet kritiske risici, som kan true projektets succes. Hver kritisk risici fik en tilhørende plan for, hvordan risikoen kan mitigers. Ydermere blev der udregnet en risikofaktor for hver risiko for at skabe et overblik over betydningen og konsekvenserne for hver risiko.

Projektet vil ved elaborationsfasen gøre brug af en kombination mellem UP & Scrum, hvor UP bruges som metode til inddeling af projektets faser, mens Scrum benyttes i forbindelse med planlægning af projektets elaborationsfase.

Bibliografi

- [1] *Unified process model*. URL: https://commons.wikimedia.org/wiki/File:Unified_Process_Model_for_Iterative_Development.svg. (Accessed: 19.02.2021).
- [2] *Inception Phase - Georgia Tech*. URL: https://www.youtube.com/watch?v=xhRoerItOF4&ab_channel=Udacity. (Accessed: 19.02.2021).
- [3] *Elaboration Phase - Georgia Tech*. URL: https://www.youtube.com/watch?v=qX5ATfVtDaE&ab_channel=Udacity. (Accessed: 19.02.2021).
- [4] *UML Tutorial*. URL: <https://www.tutorialspoint.com/uml/index.htm>. (accessed: 16.02.2021).

KAPITEL 9

Procesdokumenter

9.1 Gruppekontrakt

Fælles forventninger

1. Ærlighed samt åbenhed:
 - 1.1. Sig til gruppen, hvis man ikke har udført opgaven.
 - 1.2. Sig til i god tid hvis du har problemer, og bed om hjælp.
 - 1.3. Man bliver ikke "dømt" bare fordi man ikke har fuld forståelse.
 - 1.4. Der findes ikke dumme spørgsmål - vær ærlig og stil spørgsmål, hvis du i tvivl.
2. Der skal holdes en pæn og sober tone, men der er også plads til konstruktiv kritik.
3. Der stræbes efter at gruppearbejde bliver en god, sjov, hyggelig samt lærerig oplevelse for alle.
4. Gruppen stræber efter at være tilfredse med deres egen og fælles indsats.

Fremmøde og udeblivelse

1. Man skal informere gruppen ved manglende eller for sent fremmøde:
 - 1.1. Man skriver hurtigst mulig, hvis man er forhindret i at møde til tiden.
2. Gruppen informeres over messenger-gruppen:
 - 2.1. Hvis et medlem ved, at man ikke har mulighed for at møde, informeres resten af gruppen om dette så snart medlemmet ved besked.
3. Gruppen informeres om fravær over Discordi kanalen fravær, hvis fraværet vides mere end én dag før, ellers over messenger-gruppen:
 - 3.1. Sover et medlem over sig, svarer man så snart man vågner.

- 3.2. Hvis man af personlige årsager ikke kan deltage i gruppeaktiviteterne, informeres den resterende del af gruppen herom. Eller mindst ét andet medlem.
4. Der bruges ikke akademisk kvarter.
5. Andre tilfælde:
 - 5.1. Ved mistanke om COVID-19-smitte, følget pågældende gruppemedlem myndighedernes retningslinjer. Den resterende del af gruppen tager passende forholdsregler, så smitten ikke spredes.
 - 5.2. Ved særlige personlige tilfælde (Dødsfalder i nær familie mm.) vil medlemmets opgaver blive revideret for den følgende periode. Ved smitte informerer det pågældende gruppemedlem resten af gruppen over Messenger gruppen.

Konsekvenser ved brud på gruppekarakteren

1. Ved manglende kommunikation og udeblivelse til gruppemøder og/eller arbejdsdage, tages der en kammeratlig samtale med pågældende gruppemedlem og en adfærdsændring påkræves.
2. Ved gentagne manglende kommunikation og udeblivelse til gruppemøder og/eller arbejdsdage, vurderes det i gruppen om, hvorvidt vejleder skal involveres.
3. Ved brud af gruppekarakteren, tages der en kammeratlig samtale med pågældende gruppemedlem, og en adfærdsændring påkræves.
4. Ved gentagne brud af gruppekarakteren, vurderes det i gruppen om, hvorvidt vejleder skal involveres.

Frafald under projektet

1. Hvis et gruppemedlem vælger at stoppe på studiet, meldes det til gruppen hurtigst muligt. Frafaldet skal ydermere informeres til vejleder og uddannelseskoordinator.

Gruppearbejdsdag

1. Tirsdag er fastsat som officiel arbejdsdag, hvor der mødes klokken 10:00-18:00 medmindre andet bliver angivet:
 - 1.1. Ændringer oplyses i messenger-gruppen, hvis det er mindre end én dag før planlagt arbejdsdag.
 - 1.2. Ændringer oplyses i Discord-gruppen i kanal general, hvis der er mere end én dag før.
2. Hvis man ved, at man kommer for sent, skal man skrive en besked til gruppen.

3. Besked gives i overensstemmelse med reglerne under Fremmøde og udeblivelse.
4. Ved gruppearbejde er der fokus på projektet, selvfølgelig tillades privat snak i pausetiden.
5. Et hvert gruppemedlem kan kræve fokus på projektet, hvis urelateret snak tager overhånd.

9.2 Vejlederkontrakt

Vejlederkontrakt og aftale med vejleder

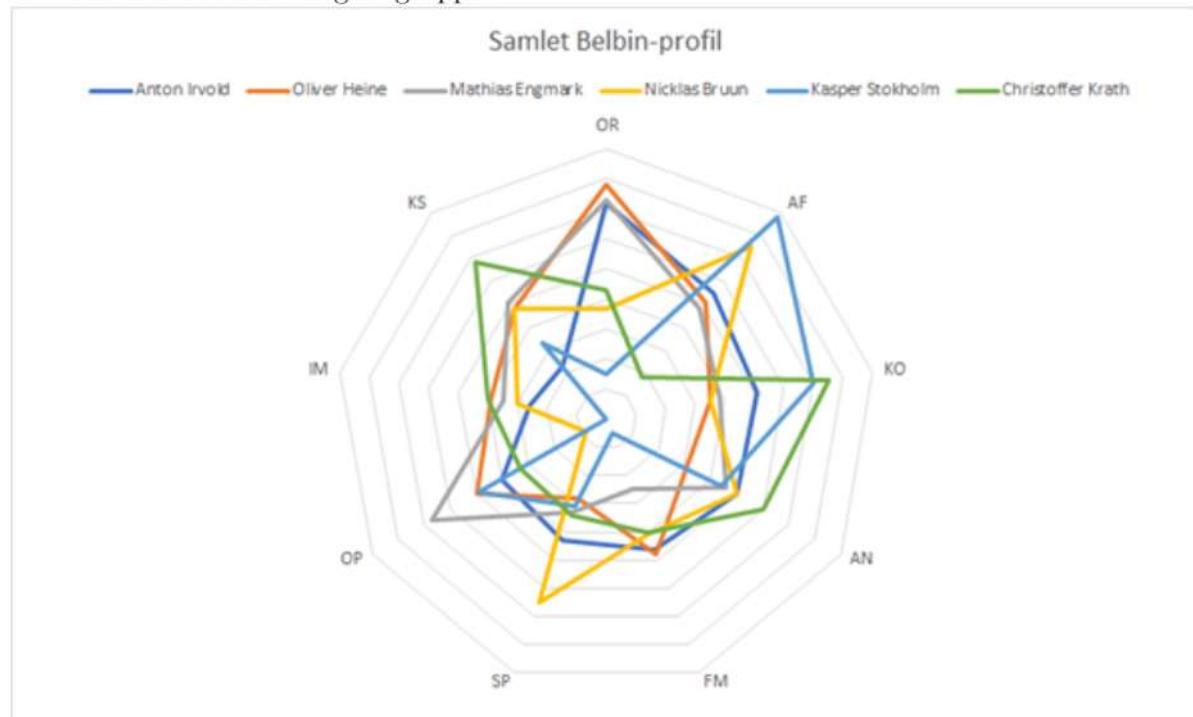
1. Alle vejledninger skal have en ordstyrer samt en referent.
2. Alle vejledermøder skal dokumenteres i logbogens WIKI-side.
3. Referenten skriver sit navn på referatet i logbogen. Referent titlen går på runde.
4. Vejledermøder foregår på MS Teams ved at Vejleder ringer op når denne er klar.
5. Som udgangspunkt er der vejledning hver tirsdagklokken 08:15.
6. Der kan godt arrangeres flere vejledermøder efter behov, men som hovedregel arrangeres der ikke færre. Hvis I ønsker at springe et vejledermøde over, så skal I have en god grund til det.
7. Vejledertiden kan variere fra gang til gang alt efter behovet, men som udgangspunkt er hver vejledning 45 minutter.
8. Flere / færre vejledermøder planlægges efter behov.
9. Materiale der ønskes feedback på til vejledermødet tirsdag morgen, skal vejleder modtage fredagen forinden, senest kl. 19:00 på mail: hela@mmmi@sdu.dk.
10. Vær specifik omkring hvad der skal gives feedback på.
11. Agenda til vejledermødet tirsdag skal fremsendes inden tirsdag morgen kl. 8:00.

Vejleder generelt

1. Materiale til Henrik som der skal gives feedback på skal sendes på mail: hela@mmmi@sdu.dk med forventet svar på max 72 timer
 - 1.1. Vær specifik omkring hvad der skal gives feedback på.
2. Mails til vejleder sendes cc til hele gruppen medmindre der er indhold af særlig følsom karakter.
3. Vejleder skal svare pr. mail inden for 72 timer.
4. Ved sending af mail, skal emne indeholde gruppenavn (Gruppe ST01)
5. Henrik kan om nød kontaktes på sms: + 45 28 94 36 02

9.3 Belbin-Teamprofil

Gruppen har lavet en samlet Belbin teamprofil for gruppen. Dette er baseret på de individuelle Belbin-profiler. Alle, undtagen Kasper der ikke havde en gruppe sidste semester, har fået feedback fra tidligere grupper.



Gruppen tager de maksimale værdier fra hver kategori, og laver en profil for denne, som viser styrker og svagheder i gruppen:



Svagheder

Profilen viser, at gruppen har en meget bred profil, men specielt mangler idemand (IM) og formidler (FM). Dette betyder, at gruppen skal være opmærksom på følgende svagheder:

- Gruppen kan have problemer med at komme med at løse vanskelige problemer.
- Gruppen kan have problemer med at generere de indledende ideer.
- Gruppen kan have problemer med at formidle vores produktidé og projekt til eksempelvis vejleder og virksomhed.

Desuden har gruppen overvægt i organisator (OR), afslutter (AF) og koordinator (KO). Dette kan betyde, at vi skal være opmærksom på, at følgende kan fylde for meget i gruppens arbejde:

- Gruppemedlemmer kan være uflexible og stædige.
- Gruppen kan have for fokuserede og unødvendige bekymringer.
- Gruppemedlemmerne kan være tilbøjelige til at manipulere.

Styrker

Gruppen har overvægt i OR, AF og KO, analysator (AN) og kontaktskaber (KS). Derfor har gruppen sandsynligvis følgende styrker:

- Gruppen har stærke organiseringsevner, og god struktur i arbejdet. Gode til at omsætte ideer til handling
- Gruppen er gode til at finde fejl og finpudse arbejdet
- Gruppen er god til at identificere talent og deleger arbejdet efter dette
- Gruppen er god til at se muligheder og arbejde strategisk og objektivt
- Gruppen er udadvendt og udforsker muligheder

Action points

- Gruppen kan have problemer med at finde ideer. Det er derfor vigtigt, at gruppen udnytter brainstormmetoder.
- Gruppen kan have problemer med at formidle produkt og projekt til vejleder og virksomhed. Det er derfor vigtigt, at gruppen forbereder pitch og agenda.
- Gruppen kan være ufleksible og stædige, samt tilbøjelige til at manipulere. Det er derfor vigtigt, at gruppen aktivt prøver at beholde et godt socialt samarbejde internt.

Appendices

Bilag

A	Checkliste for inceptionsdokument	31
B	Logbog	33
C	Referat fra reviewmøde	33

A Checkliste for inceptionsdokument

Indhold	Kriterier for godkendelse	Opfyldt +/- - /i
I. Forside	Projekttitel, uddannelsesinstitution, fakultet, institut, ud-dannelse, semester, kursuskode, projektperiode, vejleder, projektgruppe og projektdeltagere (fornavn, efternavn, sdu-email). Må gerne have illustrationer.	+
II. Titelblad	Samme oplysninger som på forsiden, samt projektdelta-gernes underskrifter (Projektdeltagernes aktive deltagelse i projektforløbet anerkendes gennidigt ved projektdeltager-nenes underskrifter. Må ikke have illustrationer.	+
III. Indholdsfor-tegnelse	Samlet indholdsfortegnelse for hele projektrapporten. Højst to eller tre niveauer i indholdsfortegnelse, der kan evt. være flere i selve rapporten. Afsnit på niveau 1 og 2 skal være nummererede.	+
1. Indledning	Projektets rammer og baggrunden for projektet. Formål med og mål for inceptionsfasen. Problemanalysen. Problem-formulering og afgrænsninger.	+
2. Faglig videns-grundlag	Begrebsdefinitioner, teori og fagligt vidensgrundlag. Rele-vante eksisterende løsninger.	+
Hovedtekst		
3. Overordnet kravspecifikation	Overordnet kravspecifikation med: <ul style="list-style-type: none"> • Overordnet brugsmønstermodel: Afgrænsning, brugsmønsterdiagram, aktørbeskrivelser, korte brugsmønsterbeskrivelser, samt detaljeret beskrivelse af udvalgte, essentielle brugsmønstre. • Overordnet beskrivelse af supplerende krav 	+
4. Kritiske risici	Identificerede kritiske risici, dvs. risici som vil være en trus-sel mod projektets succes, hvis de ikke bliver afværget: <ul style="list-style-type: none"> • Er alle kritiske risici identificeret. • Er de identificerede risici blevet afværget, eller er der en plan for at afværge dem? 	+
5. Prioritering	Prioritering der bygger på kravene forretningsmæssige be-tydning (nytte/benefit), arkitektoniske betydning, risiko samt læringsmæssige udbytte.	+
6. Metode i el-aborationsfasen	Metode, der vil blive brugt i elaborationsfasen, herunder en overordnet beskrivelse af den måde som UP og Scrum vil blive kombineret i projektet.	+
7. Ressourcer	Arbejdstidsindsats uge for uge og alt i alt.	+
8. Konklusion	Opsumering af resultaterne og diskussionen af dem. Svar på om formålet og målene med inceptionsfasen er opnået.	+
Procesdokumenter		

Appendices

1. Procesdokumenter	<ul style="list-style-type: none">● Gruppekontrakt, opdateret.● Vejlederkontrakt opdateret.● Belbin gruppeprofil.	+
Bilag		
A. Logbog	Link til logbogen	+
B. Review	Referat af review	+
C. Checkliste for inceptionsdokument	Udfyldt checkliste	+

B Logbog

Der er udført logbog for projektets forløb, hvilken kan findes på følgene link:
<https://github.com/mathias852/2.Semesterprojekt/wiki>

C Referat fra reviewmøde

Review fra Jeppe Holst Larsen Overordnede pointer:

- God opsætning gennem hele dokumentet
- Der mangler beskrivelse af supplerende krav

Individuelle pointer til afsnit: Indledning

- • Ser generelt godt ud
- Kan inkludere problemtræ i analyse, for at hjælpe med afgrænsning
- Brainstorm over ideer af virker ikke helt vigtig, da det bare minder om opstilling af krav fra TV2.

Fagligt vidensgrundlag:

- • Der mangler afsnit om tverfaglig viden.
- Skal give bedre forståelse for problemet.
- Kreditering og TV2 info mangler.
- Hører UML afsnittet med? Afhænger af hvem der skal læse rapporten.

Overordnet kravspecifikation:

- Mangler muligvis økonomiafdeling som aktør
- Mangler lidt consistency mellem brugsmønster og detaljerede brugsmønstre

Kritiske risici:

- Ser godt ud, overskueligt.

Prioritering:

- S0X krav kan ikke findes andre steder i rapporten – Supplerende krav?

Metode i elaborationsfasen:

- God forklaring af metode, ingen kommentarer.

Konklusion

- God konklusion.

Review fra Oliver Dahl Wagner Overordnede pointer:

- God opsætning, dog undgå hængende overskrifter.

Individuelle pointer til afsnit Indledning:

- Meget god indledning, kommer fint rundt.
- Rød tekst?

Fagligt vidensgrundlag:

- Mangler generelt underoverskrifter mellem kapitel og første afsnit.
- Punkt 2.2 – Generelt giver det ikke mening at skrive at gruppen har bestået 1. semester softwareteknologi.

Overordnet kravspecifikation:

- Punkt 3.1.4 (brugsmønsterdiagrammet) måtte gerne ligge tidligere i afsnittet.

Kritiske risici:

- Ser godt, overskueligt.
- Bogen har også en anden måde at sætte det op, men det ser fint ud.

Prioritering:

- Fin prioritering, måske for mange Must Haves.

Metode i elaborationsfasen:

- God forklaring af metode, ingen kommentarer.
- God forklaring af UP og SCRUM.

Konklusion:

- God konklusion - ikke meget.

Review fra Albert Gejr Nielsen Overordnede pointer:

- Forside, titelblad, indledning, fagligt vidensgrundlag er som de skal være.

Individuelle pointer til afsnit: Brugsmønsterdiagram:

- Overvej at bruge abstrakte aktører - eksempelvis en superbruger for systemadministrator og producer.

- Kan gøre diagram mere overskueligt.

Overordnet kravspecifikation:

- Godt.

Kritiske risici:

- Ser godt ud, overskueligt.

Prioritering:

- Super god opstilling.
- Krav skal bygge på forretningsmæssige benefits, måske noget at tage til overvejelse?
Hvorfor er krav prioriteret som de er. (benefit analyse af prioriterede krav)

Metode i elaborationsfasen:

- En mere konkret beskrivelse af blandingen af UP og SCRUM, kunne være med fokus på artefakter, events og en blanding af disse.

Ressourcer:

- Gant Diagram? Blot en overvejelse.

Konklusion:

- Belbin-gruppeprofil mangler umiddelbart.

Input fra Henrik Lange (vejleder) Overordnede pointer:

- Brug 'todo' i Overleaf til at lave kommentarer (skal ikke stå i teksten)
- Rapporten er skrevet til andre der arbejder med software (eksempelvis software-studerende)

Individuelle pointer til afsnit: Fagligt vidensgrundlag

- Formuleringer kan forbedres eksempelvis i afsnittet om UML (mere kort og præcist)
- Egne holdninger skal ikke fremstå i teksten.
- Beskriv ikke UML som overordnet koncept men i stedet de enkelte relevante dia-grammer. Sekvens-diagrammer, klasse diagrammer, analyse diagrammer.
- God ide at inkludere en forklaring af Unified Process (UP), da det ikke er så anvendt i industrien.
- ”Vi studerer software teknologi på 2. semester, og har gennemført disse kurser på første semester” Kunne være en forklaring af faglig vidensgrundlag.

Brugsmønsterdiagram:

- Alt man har lært at bruge kan bruges.
- Følg undervisningen fremfor vejlederen, hvis der er forvirring
- Fremvis hvordan de forskellige brugsmønstre relaterer sig til hinanden.

Kritiske risici:

- Risikofaktor afslører de største risici, når sandsynlighed og alvorlighed ganges.

Prioritering:

- Eventuelt spørg Lone hvad hendes holdning til mængden af must have krav.
- Virksomhedseksempel med 300 krav har 93% must have krav, og det vil ikke komme under 50% must have krav.

Ressourcer:

- GANT kan bruges i Inceptionsfasen (ikke i kombination af SCRUM, da det er det modsatte)