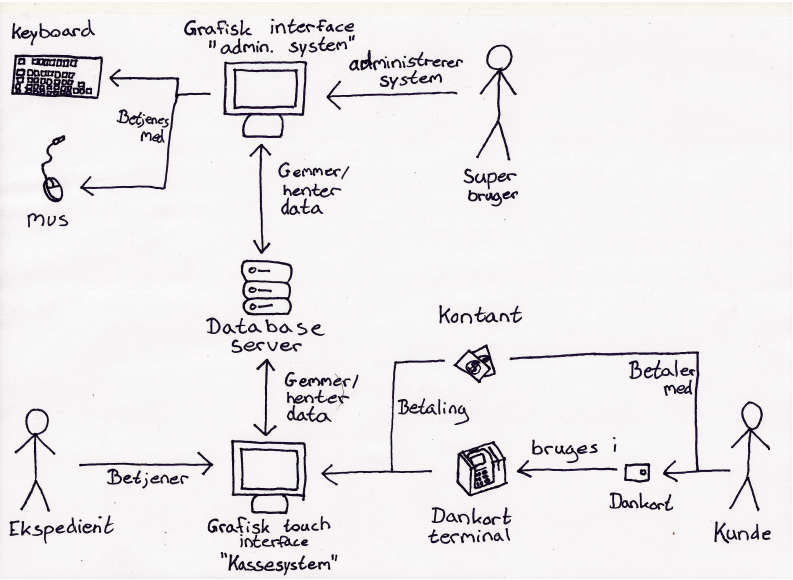
Salgssystem

Projektdokumentation

Aarhus Universitet
Gruppe 3 - 4. Semester - E15
XX. dec 2015

Kasper Klausen	0000000
Benjamin Larsen	0000000
Dennis Tychsen	0000000
Martin Carpentier	0000000
Jakob Kristensen	0000000
Andreas Engelbrecht	0000000



Vejleder - Lars Mortensen

Gruppe 4

Kasper Klausen | 201303660	Dato
Benjamin Larsen | 201303646	Dato
Dennis Tychsen | x	Dato
Martin Carpentier | x	Dato
Jakob Kristensen | x	Dato
Andreas Engelbrecht | x	Dato

Indhold

0.1 Arbejdsfordelings Liste

Oversigt over hvem der har været med til at implementere de forskellige dele

Generelt		
Opgave	**Primær**	**Sekundær**
Kravspecifikation	Alle	
Accepttestspecifikation	Alle	

Kasseapparat		
Opgave	**Primær**	**Sekundær**

Administrationssystem		
Opgave	**Primær**	**Sekundær**

CentralServer		
Opgave	**Primær**	**Sekundær**

SharedLib		
Opgave	**Primær**	**Sekundær**

1 Kravspecifikation

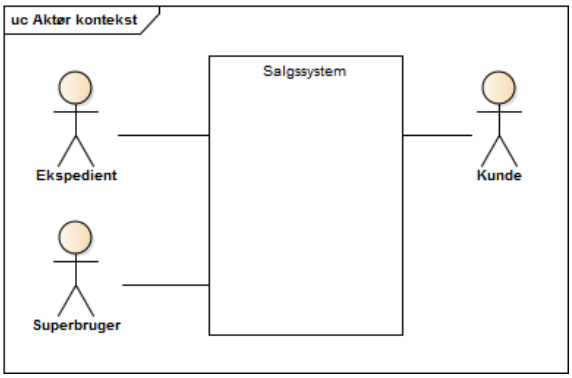
1.1 Revision

1.2 Systembeskrivelse

Vi fremstiller et salgssystem til fysiske butikker. Systemet består af en central server, hvortil et eller flere kasseapperater er tilsluttet. Ekspedienter kan ekspedere kunder via et kasseapperat, som kan tilgås via en skærm eller touch-skærm. Superbruger kan tilgå et administrationssystem igennem en vilkårlig maskine, som har adgang til internettet eller på lokalt netværk med Central server. Her er der mulighed for at oprette, redigere og slette produkter og produktgrupper.

Den centrale server indeholder en database, hvor produkter, produktgrupper mv. ligger gemt. Herfra henter kasseapperaterne de nødvendige informationer såsom produktoplysninger. Kasseapperater informerer desuden den centrale server når salg og andre relevante begivenheder finder sted.

1.3 Aktører



Figur 1.1: Aktør kontekst diagram for salgssystemet.

Det ses hvordan både Ekspedient og Supahburger er de primære aktører for systemet. På baggrund af aktør kontekstdiagrammet i figur 1.2, er følgende aktørbeskrivelse udarbejdet.

Navn	Ekspedient
Type	Primær
Beskrivelse	Ekspedient er en primær bruger af systemet, som benytter systemet til at sælge og returnere vare.

Tabel 1.1: Aktør beskrivelse af Ekspedient

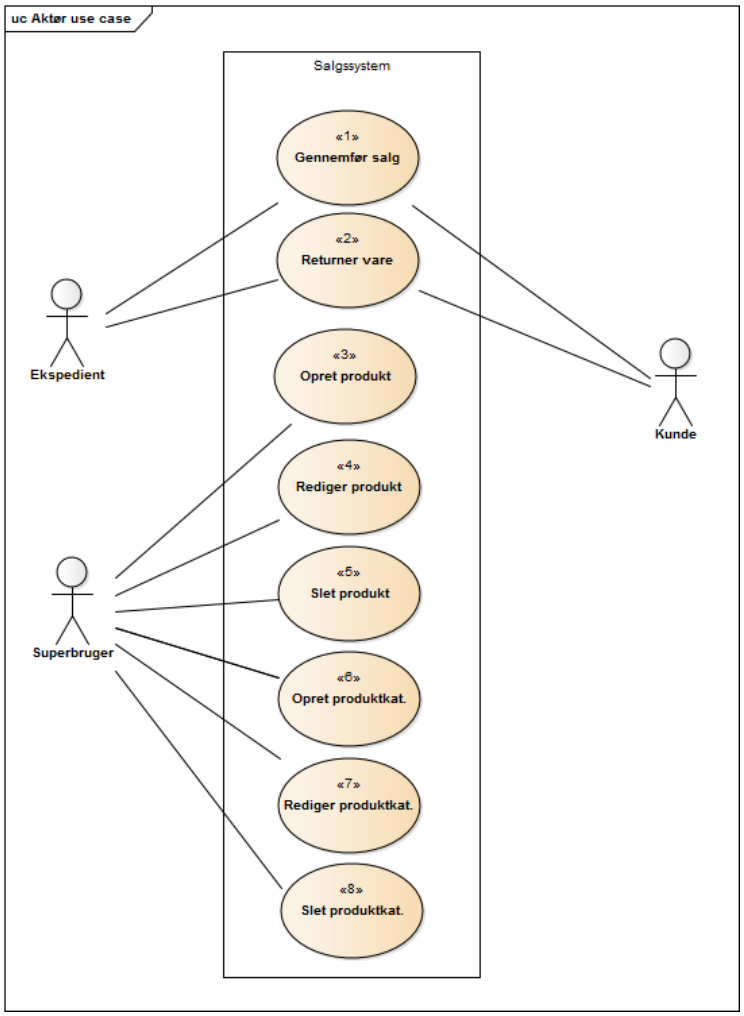
Navn	Super User
Type	Primær
Beskrivelse	Supahburger er en primær bruger af systemet, som benytter systemet til at oprette, nedlægge og redigere i systemets varekatalog.

Tabel 1.2: Aktør beskrivelse af Supahburger

Navn	Kunde
Type	Sekundær
Beskrivelse	Kunden er en sekundær bruger af systemet, og ønsker at købe/returnere en eller flere varer igennem Ekspedient.

Tabel 1.3: Aktør beskrivelse af kunden

1.4 Aktør Use Case Diagram



Figur 1.2: Aktør usecase diagram for salgssystemet.

I figur 1.2 ses det hvordan de forskellige aktører har forbindelse med de individuelle usecases.

1.5 Funktionelle Krav - Use Cases

MoSCoW

Systemet *skal* kunne

- Systemet skal have et Administrationssystem, hvor en Supahburger kan oprette, redigere og slette Produktkategori og Produkt i databasen
- Systemet skal indeholde et Kasseapparat, hvor . . . :
  - En Ekspedient kan sælge produkter til kunder
  - En kunde kan returnere varer til en Ekspedient
- Kasseapparat skal . . . :
  - Kunne tage imod kontant betaling
  - Danne en salgskvittering
  - Danne en rekvisionskvittering
- Indeholde en Database, som indeholder et produktkatalog

- Indeholde kommunikation mellem Administrationssystem og Kasseapparat

Systemet *bør* kunne

- Understøtte brugerlogin i flere niveauer
  - En Supahburger, der kan tilgå Administrationssystem
  - En Ekspedient, der kan tilgå Kasseapparat
- Ekspeditioner kan parkeres midlertidigt
- Benytte variabel momssats per Produkt
- Understøtte flere forskellige instanser af Kasseapparat på samme tid
- Understøtte tilbudskampagner
- Have kontrol af lagerbeholdningen i Administrationssystem
- En Ekspedient kan afskrive et Produkt, hvis den f.eks. er gået i stykker
- Vise salgsstatistik i Administrationssystem
- Udskrive en salgskvittering via Kasseapparat
- Kasseapparat bør kunne tage imod dankort betaling

Systemet *kunne* evt.

- Understøtte stregkodescanner
- Understøtte betaling med forskellige slags valuta
- Have mulighed for, at kunder, som har en konto, kan få rabat
- Tage imod betaling fra en mobilapplikation
- VIP Medlemmer skal kunne modtage en email med deres salgskvittering

Use case 1: Gennemfør salg

Navn	Gennemfør salg
Formål	Gennemføre et salg af et vilkårligt antal varer til Kunde
Initialisering	Ekspedient
Aktører	Ekspedient (Primær), Kunde (Sekundær)
Reference	Ingen
Samtidige forekomster	Én per instans af Kasseapparat
Prækondition	Kasseapparat har forbindelse til CentralServer
Postkondition	Kunde har købt sine varer og salget er gemt i Database
Hovedscenarie	1. For hver vare Kunde vil købe: a. Ekspedient indtaster vare b. Ekspedient vælger vare og indtaster antal og vælger Antal [Ext 1: Forkert vare indtastet] [Ext 2: Forkert antal indtastet] [Ext 3: Handlen afbrydes] 2. Kunde giver Ekspedient penge 3. Ekspedient indtaster beløbet, som Kunde har givet 4. Ekspedient trykker på knappen til valgte betalingsform 5. Kunden får eventuelt restbeløb tilbage
Extensions	[Ext 1: Forkert vare indtastet] 1. Ekspedient vælger varen på liste over vare 2. Ekspedient trykker på delete knappen [Ext 2: Forkert antal indtastet] 1. Ekspedient vælger varen på liste over vare 2. Ekspedient indtaster det korrekte antal 3. Ekspedient trykker på “Antal” for at bekræfte [Ext 3: Handlen afbrydes] 1. Ekspedient trykker på annuller knappen

Tabel 1.4: Use case 1: Gennemfør salg

Use case 2: Returner vare

Navn	Returner vare
Formål	Returnere en fejlkøbt vare
Initialisering	Ekspedient
Aktører	Ekspedient (Primær) Kunde (Sekundær)
Reference	Ingen
Samtidige forekomster	Én per Kasseapparat
Prækondition	Kasseapparat har forbindelse til CentralServer
Postkondition	Kunde har returneret sin vare og fået tilsvarende penge udbetalt
Hovedscenarie	1. Ekspedient indtaster varen 2. Ekspedient vælger vare og indtaster antal og vælger Retur [Ext 1: Kunde fortryder] 3. Ekspedient indtaster 0 + kontant 4. Ekspedient giver Kunde angivet beløbet tilbage
Extensions	[Ext 1: Kunde fortryder] 1. Ekspedienten vælger Annuller

Tabel 1.5: Use case 2: Returner vare

Use case 3: Opret Produkt

At oprette en Produkter til systemet.

Navn	Opret Produkt
Formål	Supahburger opretter Produkt .
Initialisering	Supahburger
Aktører	Supahburger (Primær)
Reference	Ingen
Samtidige forekomster	∞
Prækondition	Der er oprettet forbindelse til CentralServer.
Postkondition	Der er oprettet et nyt, tidligere ubenyttet Produkt i systemet.
Hovedscenarie	1. Supahburger trykker på opret Produkt på i Administrationssystem. 2. Et pop up vindue åbnes i GUI, hvori der indtastes informationer om Produktet. 3. Supahburger trykker på gem. [Ext 1: Produkt eksisterer allerede] 2. Systemet sender kommando til CentralServer. [Ext 2: CentralServer melder fejl. 3. Administrationssystem modtager opdatering om nyt Produkt. 4. Administrationssystem opdaterer Produktlisten
Extensions	[Ext 1: Produktet eksisterer allerede] 1. Beskedboks informerer Supahburger om at Produkt allerede eksisterer. 2. Der fortsættes fra punkt 1. [Ext 2: CentralServer melder fejl] 1. Beskedboks informerer Supahburger om fejl. 2. Usecasen afsluttes

Tabel 1.6: Use case 3: Usecase for opret Produkt

Use case 4: Rediger Produkt

Navn	Rediger Produkt
Formål	Produkt bliver redigeret
Initialisering	Supahburger
Aktører	Supahburger
Reference	Ingen
Samtidige forekomster	∞
Prækondition	Der er oprettet forbindelse til CentralServer
Postkondition	Det valgte Produkt er opdateret med ny data
Hovedscenarie	1. Supahburger vælger det Produkt der ønskes redigeret, og trykker på knappen til at redigere Produkt. 2. Et nyt vindue åbner, med det valgte Produkts nuværende informationer. 3. Supahburger redigerer det data der ønskes ændret og trykker på knappen til gem. [Ext 1: Supahburger vælger at annullere ændringen] [Ext 2: CentralServer melder fejl] 4. Vinduet lukker og det valgte Produkt opdateres med det nye data.
Extensions	[Ext 1: Supahburger vælger at annullere ændringen.] 1. Supahburger trykker på knappen til at annullere. 2. Vinduet til at redigere data lukker, hovedmenuen vises og det valgte Produkt forbliver uændret. [Ext 2: CentralServer melder fejl] 1. Beskedboks informerer Supahburger om fejl. 2. Usecasen afsluttes

Tabel 1.7: Use case 4: Rediger Produkt

Use case 5: Slet Produkt

Navn	Slet Produkt
Formål	At slette et Produkt
Initialisering	Supahburger
Aktører	Supahburger
Reference	Ingen.
Samtidige forekomster	∞
Prækondition	Der skal være et Produkt at slette.
Postkondition	Det valgte Produkt skal være slettet.
Hovedscenarie	1. Supahburger vælger det Produkt der ønskes slettet og trykker på knappen til at slette Produkt. 2. Et dialogvindue, der spørger om Produktet ønskes slettet, vises på skærmen. 3. Supahburger trykker på knappen til at godkende sletningen og dialogvinduet forsvinder. [Ext 1: Supahburger vælger at annullere sletningen.] 4. Produktet slettets umiddelbart herefter fra listen over Produkter. [Ext 2: CentralServer melder fejl]
Extensions	[Ext 1: Supahburger vælger at annullere sletningen.] 1. Supahburger trykker på knappen til at annullere sletningen. 2. Dialogvinduet forsvinder og intet yderligere foretages. [Ext 2: CentralServer melder fejl] 1. Beskedboks informerer Supahburger om fejl. 2. Usecasen afsluttes

Tabel 1.8: Use case 5: Slet Produkt

Use case 6: Opret Produktkategori

Navn	Opret Produktkategori
Formål	At oprette en Produktkategori til Produkter
Initialisering	Supahburger
Aktører	Supahburger (Primær)
Reference	Ingen
Samtidige forekomster	∞
Prækondition	Der er oprettet forbindelse til CentralServer og der er instanser af Produktkategori i systemet.
Postkondition	Der er oprettet en *ny* Produktkategori i systemet
Hovedscenarie	1. Supahburger trykker på knappen til opret Produktkategori 2. Et pop up vindue åbnes i GUI, hvori der indtastes informationer om Produktkategori. 3. Supahburger trykker på knappen til gem [Ext 1: Produktkategori eksisterer allerede] 3. Systemet sender kommando med information til CentralServer. [Ext 2: CentralServer melder fejl. 4. Administrationssystem modtager opdatering om ny Produktkategori. 5. Administrationssystem opdaterer kategorilisten
Extensions	[Ext 1: Produktkategori eksisterer allerede] 1. Beskedboks informerer Supahburger om at Produktkategori eksisterer allerede. 2. Der fortsættes fra punkt 2. [Ext 2: CentralServer melder fejl] 1. Beskedboks informerer Supahburger om fejl. 2. Usecasen afsluttes

Tabel 1.9: Use case 6: Opret Produktkategori

Use case 7: Rediger Produktkategori

Navn	Rediger Produktkategori
Formål	At redigere en Produktkategori
Initialisering	Supahburger
Aktører	Supahburger (Primær)
Reference	Ingen
Samtidige forekomster	∞
Prækondition	Forbindelse til CentralServer er oprettet
Postkondition	Produktkategorien er redigeret i systemet
Hovedscenarie	1. Supahburger trykker på den Produktkategori der ønskes at redigere i listen af Produktkategorier. 2. Supahburgeren trykker på knappen til rediger Produkt i hovedmenuen. 3. Supahburgeren ændrer de ting der ønskes at redigeres, og trykker på knappen til gem. 4. Administrationssystem modtager opdatering om redigeret Produktkategori. [Ext 1: CentralServer melder fejl. 5. Administrationssystem opdaterer listen af Produktkategorier
Extensions	[Ext 1: CentralServer melder fejl] 1. Beskedboks informerer Supahburger om fejl. 2. Usecasen afsluttes.

Tabel 1.10: Use case 7: Rediger Produktkategori

Use case 8: Slet produktkategori

Navn	Slet produktkategori
Formål	At slette en produktkategori
Initialisering	Superbruger
Aktører	Superbruger (Primær)
Reference	Ingen
Samtidige forekomster	∞
Prækondition	Forbindelse til serveren er oprettet
Postkondition	Produktkategorien er slettet i systemet
Hovedscenarie	1. Superbruger trykker på den kategori der ønskes at slette i listen af kategorier. 2. Superbruger trykker på knappen til slet kategori i hovedmenuen og der vises et vindue. 4. Superbruger vælger hvilken kategori han ønsker at flytte produkterne til og trykker på knappen til at flytte produkterne. [Ext 1: Der er ingen produkter i kategorien.] [Ext 2: Superbruger annullerer sletningen] 5. Superbrugeren trykker på slet kategori, efter produkterne er flyttet. 5. Vinduet lukker og kategorien bliver slettet fra listen over kategorier. [Ext 3: Central server melder fejl] 8. Administrationssystemet opdaterer kategorilisten
Extensions	[Ext 1: Der er ingen produkter i den valgte kategori.] 1. Knappen til Flyt er grå, og kan ikke aktiveres. [Ext 2: Brugeren annullerer sletningen] 2. Usecasen afsluttes [Ext 3: Central server melder fejl] 1. Beskedboks informerer brugeren om fejl. 2. Usecasen afsluttes.

Tabel 1.11: Use case 8: Slet produktkategori

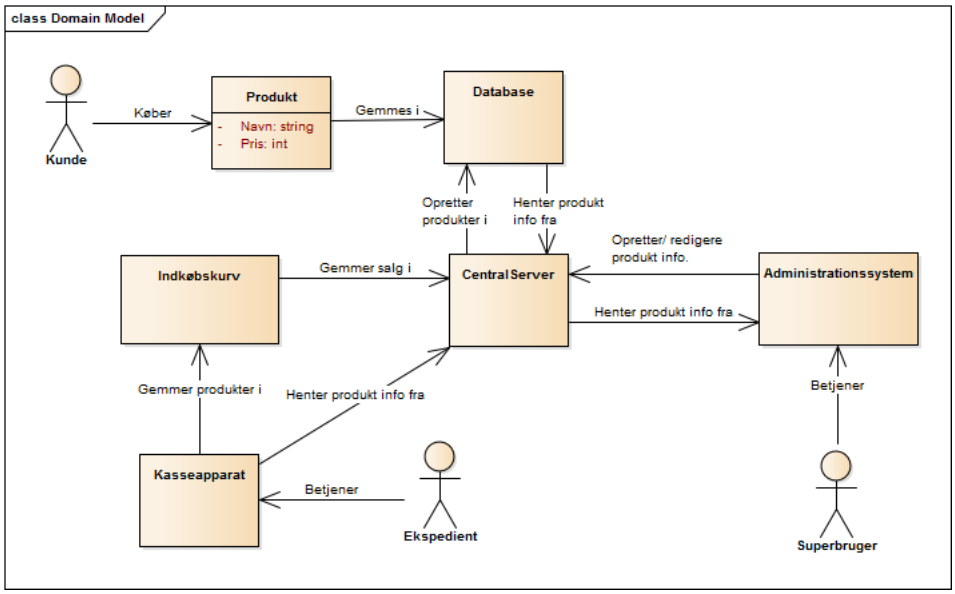
1.6 Ikke Funktionelle Krav

- Når der oprettes/redigeres/slettes et Produkt/en Produktkategori i Administrationssystem må der max. gå 15 sekunder før det fremgår i Kasseapparat
- En kvittering skal kunne hentes fra Database på under 5 sekunder.

2 Systemarkitektur

Herunder vil systemet blive beskrevet overordnet med fokus på at skabe en forståelse for den store helhed. Ved at læse systemarkitekturen igennem vil/ burde læseren få en meget bedre forståelse for hvad systemet indebærer samt de store byggeblokke. For at forstå selve byggeelementerne samt hvordan de er sat sammen, skal læseren se i systemdesignet. Her vil mere implementeringsspecifikke detaljer være beskrevet.
For at give et overblik over systemet blev domænemodellen på Figur 2.1 opsat.

Domænemodel



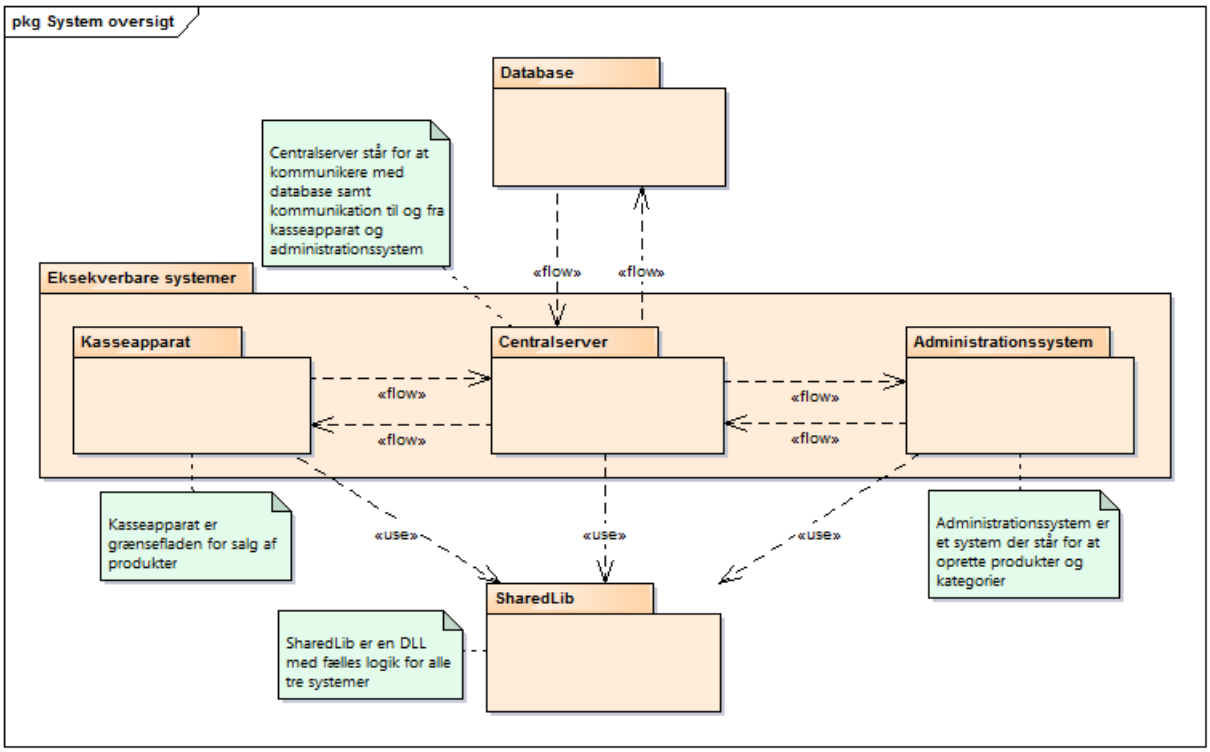
Figur 2.1: Domænemodel over systemet og dets overordnede dele

Domænemodellen på figur 2.1 viser systemets overordnede dele. En *kunde* vil købe et *produkt*. Dette *produkt* ligger gemt i *databasen*.
En *ekspedient* betjener et *kasseapparat*. Informationer om de *produkter* der er til rådighed samt deres pris hentes fra *CentralServeren*. Når en *kunde* ønsker at købe et *produkt*, så tilføjer *ekspedienten produktet* til en *indkøbskurv*. Når salget gennemføres gemmes salget i *CentralServeren*.
Produkt informationer tilføjes, redigeres eller slettes af en *superbruger* i *administrationssystemet*. Disse ændringer afspejler *CentralServeren* så i databasen.

2.1 Logisk View

Oversigt

Der er herunder opsat et pakkediagram over de delsystemer vi har indentificeret i projektet. Hver pakke symboliserer derved et system der skal implementeres individuelt. «Flow» fortæller at der er kommunikation imellem systemerne, og «use» symboliserer at systemerne benytter sig af pakken. Dybere forklaring af hvordan hver pakke er designet vil blive beskrevet i designdelen.



Figur 2.2: Oversigt over systemet i pakker

En nærmere beskrivelse af hver pakkes ansvar vil blive beskrevet på næste side under Arkitektur-pakker.

Arkitekturpakker

Arkitekturpakkerne er en beskrivelse af de pakker der ses på diagrammet under oversigten, side 20, figur 2.1. Beskrivelserne forklarer, hvad pakkens ansvar er og hvordan pakken kommunikerer med andre pakker i systemet.

SharedLib

Ansvar

SharedLib har i dette system ansvar for alt fælles logik, og bliver derfor tilgået af alle de tre delsystemer: Kasseapparat, CentralServer og Administrationssystem. SharedLib er et bibliotek indeholdende datamodeller, kommandoer og en protokol der sørger for at encode og decode disse kommandoer henholdsvis til og fra XML.

Database

Ansvar

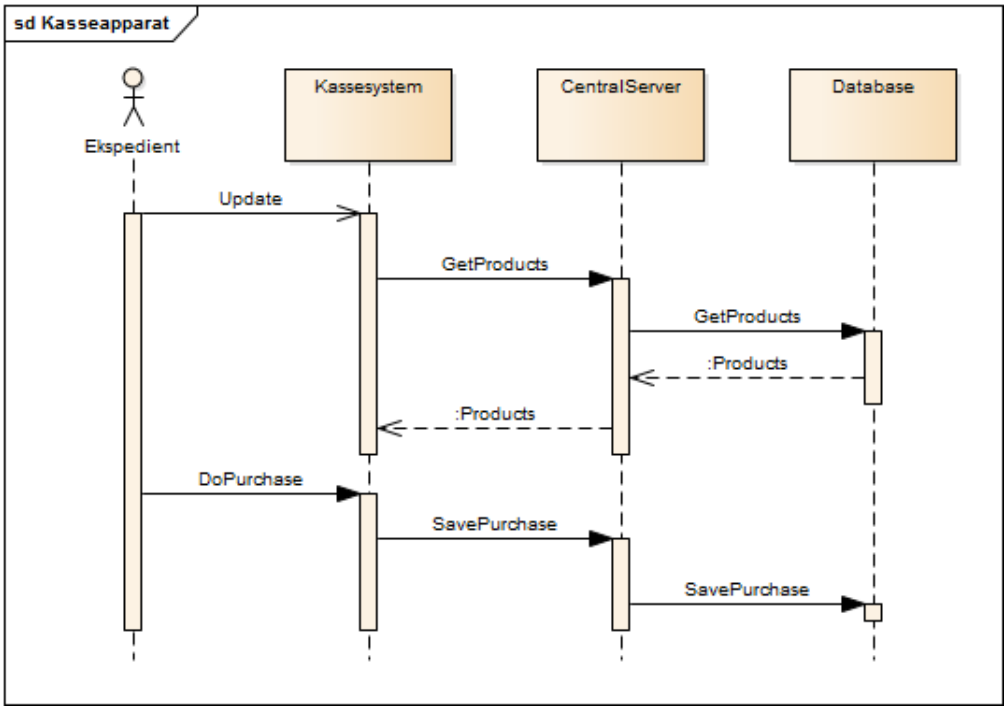
Databasen benyttes til at opbevare persistent data, herunder Produkt, Produktkategori og køb. Det er udelukkende CentralServer, som interagerer med Database.

Kasseapparat

Ansvar

Kasseapparatet er den del af systemet hvor produktkataloget bliver vist og benyttet til salg af produkter. Her laves kvitteringer og der bliver vist en liste/indkøbskurv med de produkter som en evt. kunde ønsker at købe. Produktkataloget bliver hentet ved en forespørgsel til Centralserver og kasseapparatet ved derved intet om oprettelse og redigering af de produkter som den fremviser.

Sekvensdiagram



Figur 2.3: Sekvensdiagram for kommunikation mellem Kasseapparat og andre pakker

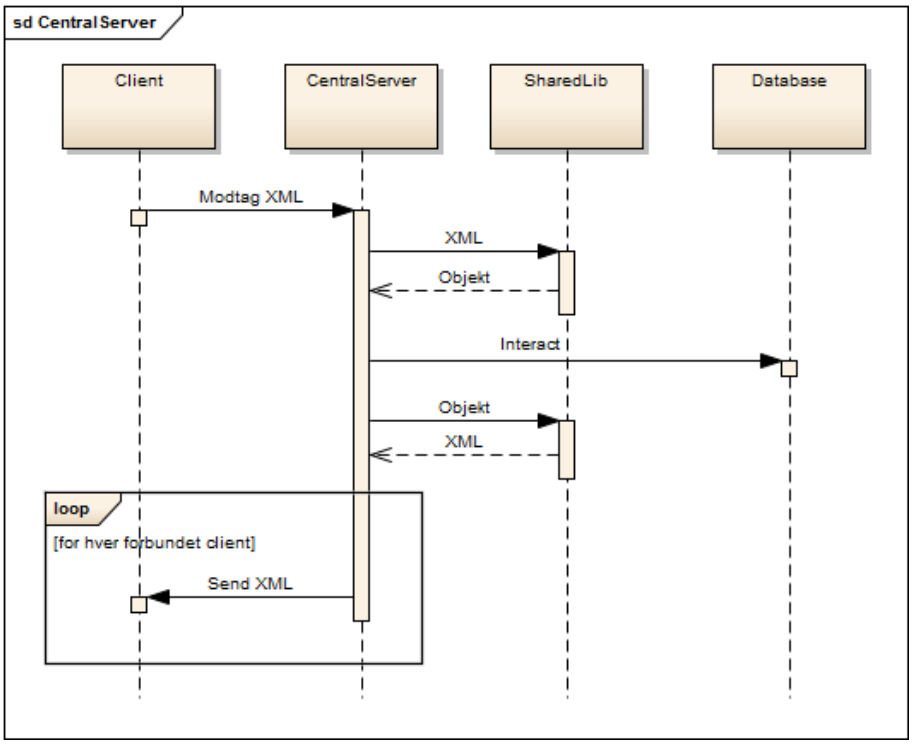
Sekvensdiagrammet viser hvordan kasseapparatet kommunikerer med de andre pakker i systemet. Kasseaparatet kommunikerer udelukkende med Centralserver. Når kasseapparatet skal sende en besked til Centralserver bruger den SharedLib til at konvertere et kommandoobjekt til XML og sender herefter denne til Centralserveren. Centralserveren sender derefter en XML-besked tilbage som så derefter bliver omskrevet til et kommandoobjekt, igen ved brug af SharedLib.

CentralServer

Ansvar

CentralServer har til ansvar at få videreformidlet kommandoer og fungerer på denne måde som mellemled imellem pakkerne. Derudover har CentralServer også ansvar for at indsætte og hente data fra databasen.

Sekvensdiagram



Figur 2.4: Sekvensdiagram for kommunikationen mellem CentralServer og andre pakker.

På figur 2.4 ses hvordan CentralServer, når den modtager XML, benytter SharedLib til at konvertere det modtagede XML til et højniveau kommando-objekt, som CentralServer herefter kan agere på. Det enten værende indsættelse/sletning af data i databasen eller andet. Til sidst vil CentralServer igen benytte SharedLib til at konverete et højniveau svar om til XML, som sendes ud til alle tilstedeværende klienter.

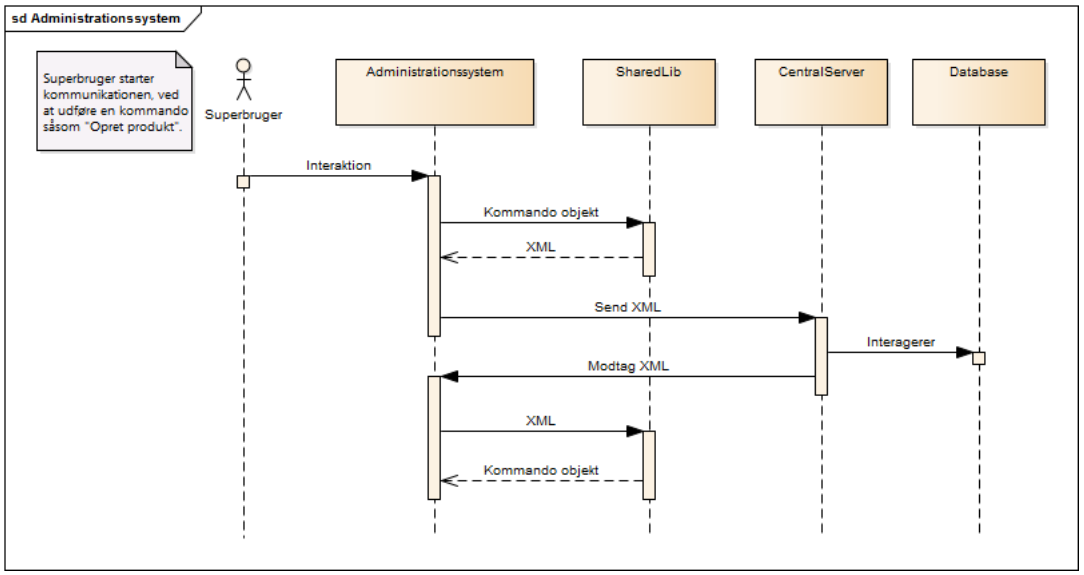
Dette er det typiske sekvens af kommunikation mellem klient og server, men det er ikke noget krav, at CentralServer agerer på den modtagede besked, eller for den sags skyld svarer tilbage til klienten.

Administrationssystem

Ansvar

Administrationssystem er den del af systemet, hvor produktkataloget bliver redigeret. Dette indebærer oprettelse, sletning og ændring af Produkt, såvel som Produktkategori. Pakken bliver betjent af Supahburger, se figur 1.2, side 8.

Sekvensdiagram



Figur 2.5: Sekvensdiagram for kommunikation mellem Administrationssystem og andre pakker

Sekvensdiagrammet på figur 2.5 viser den kommunikation der foregår mellem Administrationssystemet og de andre pakker i systemet. Administrationssystemet kommunikerer udelukkende med CentralServer, mens SharedLib bliver brugt til at encode og decode XML strenge der sendes og modtages fra CentralServer. Sekvensdiagrammet (figur 2.5) viser udelukkende interfacet mellem pakkerne. Uddybende forklaring af Administrationssystem pakken kan findes under dens design-afsnit, se afsnit 3.2.

Use Case realiseringer

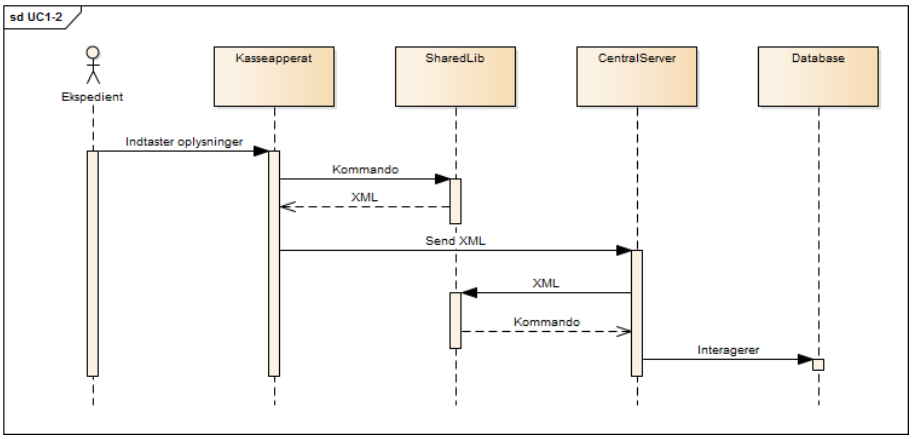
Use Case 1 - Gennemfør salg

Disse Use Cases beskriver følgende funktionalitet:

- Gennemfør salg
- Returner vare

Funktionaliteten til disse Use Cases ligner hinanden. Når Ekspedient har udført de nødvendige handlinger i Kasseapparat GUI og trykker på knappen til at afslutte handel, så bliver der sendt data til CentralServer.
Når ekspedienten har gennemført et salg/retur i Kasseapparat så vil følgende handlinger blive udført i systemet:

- Kasseapparat opretter det korrekte kommando objekt, via SharedLib, med informationerne for handlen
- Kasseapparat bruger SharedLib til at konvertere kommando objektet til en XML streng
- XML strengen sendes til CentralServer
- CentralServer modtager strengen og konverterer den tilbage til den oprindelige kommando, igen ved hjælp af SharedLib
- CentralServer udfører de nødvendige handlinger på Database



Figur 2.6: Sekvensdiagram for realisering af Use Cases 1 og 2

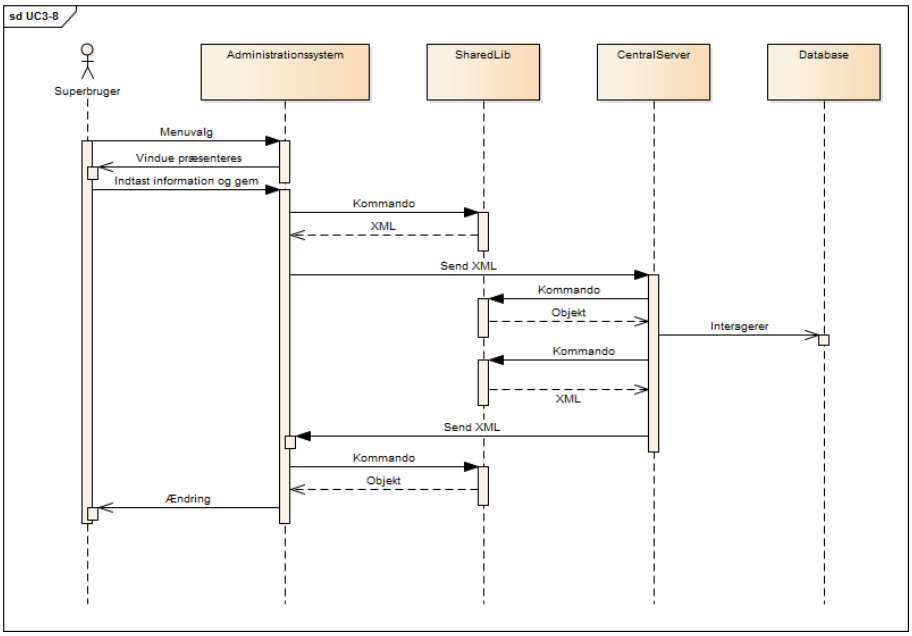
Use Cases 3-8
Disse Use Cases beskriver følgende funktionalitet:

- Opret produkt
- Rediger produkt
- Slet produkt
- Opret kategori
- Rediger kategori
- Slet kategori

Funktionaliteten til disse Use Cases ligner hinanden. Når glsSB har udført de nødvendige handlinger i Administrationssystemets grafiske brugerflade og trykker på knappen til at godkende handlingen, så bliver der sendt data til CentralServer og efterfølgende modtaget data fra CentralServer. Når Supahburger har modificeret et produkt eller en kategori i Administrationssystem, så vil følgende handlinger blive udført i systemet:

- Administrationssystem opretter et kommando objekt af den rette type, med informationerne for handlingen.
- Administrationssystem bruger SharedLib til at konvertere kommando objektet til en XML streng.
- XML strengen sendes til CentralServer.
- CentralServer modtager strengen og konverterer den tilbage til den oprindelige kommando, igen ved hjælp af SharedLib.
- CentralServer udfører de nødvendige handlinger på Database.
- CentralServer opretter et kommando objekt med det rette svar (f.eks. ProductCreated efter den har oprettet et produkt).
- CentralServer konverterer kommandoen til en XML streng ved hjælp af SharedLib.
- CentralServer sender XML strengen til alle de klienter der er forbundet til CentralServer.

Funktionaliteten for Use Casene er beskrevet med et sekvensdiagram, som vises på figur 2.7.



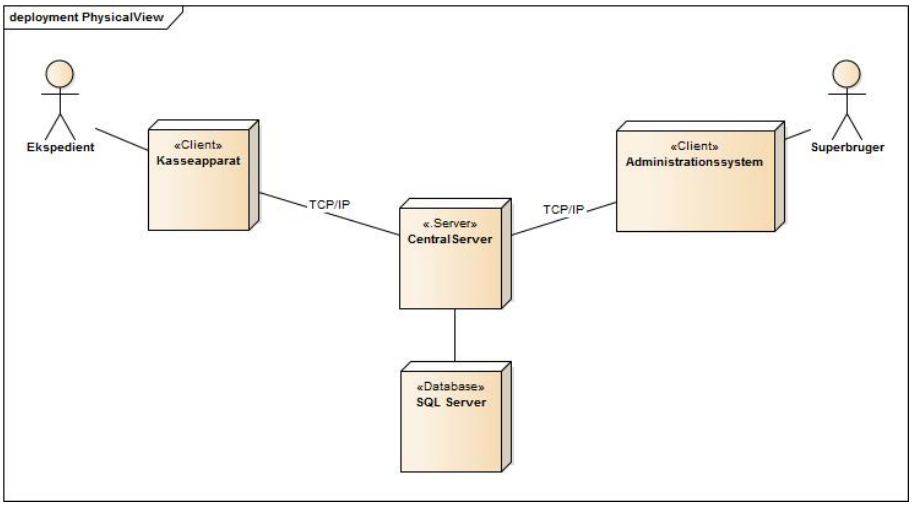
Figur 2.7: Sekvensdiagram for realisering af Use Cases 3 til 8.

Når CentralServer svarer, er der på sekvensdiagrammet (figur 2.7) afbilledet, at den blot svarer til Administrationssystemet. I realiteten broadcaster den svaret til alle klienter der er forbundet. Dette er dog ikke vigtigt for forståelsen af kommunikationen i disse Use Cases.

2.2 Physical View

Oversigt

Vi har valgt at tage Physical View med fordi det beskriver hvordan de forskellige del-systemer er forbundne, og hvilke veje kommunikationen mellem dem foregår.



Figur 2.8: Physical view af systemet

Alle del-systemer kommunikerer vha. TCP/IP [2] sockets, og systemerne kan derfor eksekvere og kommunikere asynkront på forskellige fysiske maskiner. Ekspedient og Supahburger kan derfor tilgå hvert deres system på samme tid fra forskellige maskiner.

Node Beskrivelser

Kasseapparat

Kasseapparat kommunikerer med CentralServer gennem en TCP/IP forbindelse. Forbindelsen benyttes til at hente produktkataloget samt at registrere køb.

Administrationssystem

Administrationssystem kommunikerer med CentralServer gennem en TCP/IP forbindelse. Forbindelsen benyttes til at hente produktkataloget samt at oprette, redigere og slette Produkt og Produktkategori.

CentralServer

CentralServer stiller en server til rådighed, som Kasseapparat og Administrationssystem kan forbinde til (beskrevet ovenfor). CentralServer foretager også alt den nødvendige interaktion med Database.

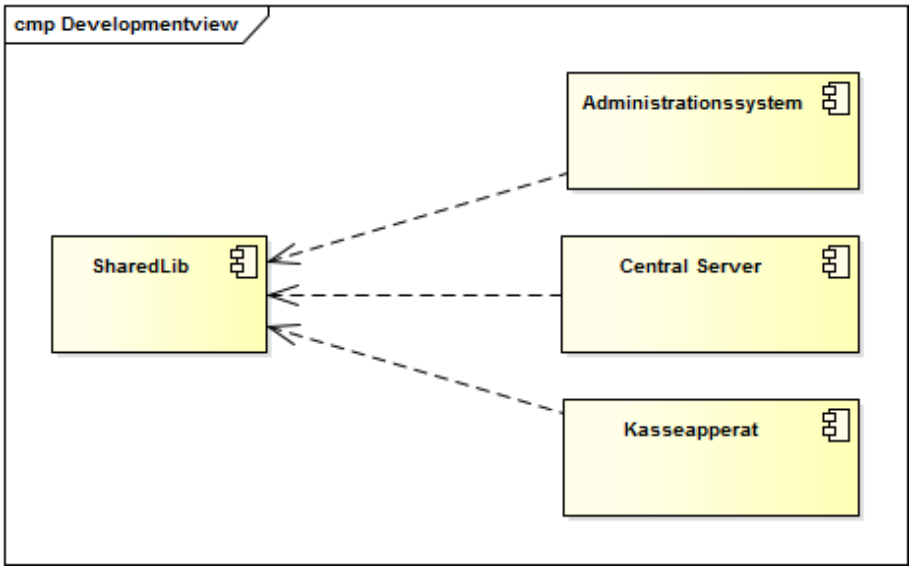
Database

Interageres med af CentralServer.

2.3 Development View

Oversigt

Der er valgt at benytte dette view til at vise afhængigheder ifm. de forskellige komponenter og noder. Dette virker yderst relevant for at kunne danne et overblik for eventuel videre udvikling.



Figur 2.9: Development view for systemet

På diagrammet i figur 2.9 ses det hvordan de tre funktionelle systemer[1] er afhængig af det delte bibliotek SharedLib. I dette bibliotek findes datamodeller og kommunikationsprotokoller, som de andre systemer bruger og er afhængige af.
En eventuel ændring i SharedLib, ville have betydelige konsekvenser for de resterende systemer. Det er derfor vigtigt at gøre sig overvejelser over hvor disse afhængigheder er i systemet, inden man foretager større ændringer. Eksempler på konkrete afhængigheder kunne være:

- Produkt
- Produktkategori
- XML-marshallere mf.

En reference må da være til stede i alle systemer, som ønsker at arbejde med produkter eller kommunikation i systemet.

Komponentbeskrivelser

Administrationssytstem

Denne bruger sharedLib i samtlige lag. Dette indebærer bl.a. modeller, protokoller, marshallers mv.

Central server

Denne benytter sharedLib til protokoller og modeller.

Kassesystemet

benytter som administrationssystemet sharedLib i dets grafiske lag samt business logic lag. Igen bruges der modeller og protokoller så alle er enige om disse.

SharedLib

Denne en fælles afhængighed for alle systemets komponenter. Denne indeholder modeller , xml-marshallers, socketforbindelser, encoder, decoders mv. s

[1] CentralServer samt Administrationssystem og Kasseapparat

3 Systemdesign

3.1 Kassesystem

Introduktion

Kasseapparatet er et program som viser produkter og kan håndtere disse i en indkøbskurv, samt herfra gennemføre salg og danne salgskvitteringer. Produkterne som vises i kasseapparatet hentes fra Centralserveren[1]. De viste produkter i kasseapparatet opdateres ikke for hver ændring i produkter i CentralServer, men når en ekspedient beder om en opdatering manuelt. Denne funktionalitet er valgt, da det ikke var ønsket at produkter kom frem og forsvandt under brug af systemet. Da dette ville være forstyrrende for en ekspedient.

Design og struktur

Kasseapparatet er kun tildels designet med MVVM [3], men dog med stor fokus på adskilelse af Forretningslag og Præsentationslaget. Dette er sikret ved brug af klasser der står for at styre logikken i grænsefladen samt kommunikere ned i de nedre lag. Derved er de dele som ikke direkte implementere MVVM dog stadig inspireret heraf.

Presentation Layer

Business Logic Layer

Data Access Layer

[1] Se beskrivelse af systempakker på figur 2.2, side 20

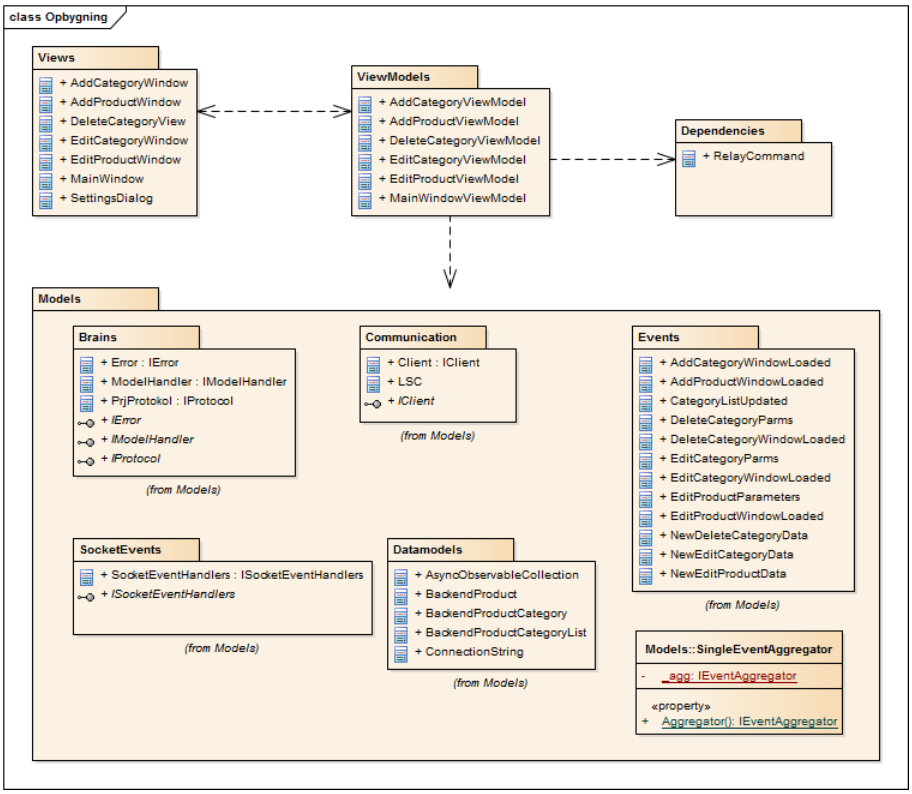
3.2 Administrationssystem

Introduktion

Administrationssystemet er en applikation som kan tilføje, fjerne og redigere produkter eller kategorier, tilhørende kassesystemets database.
Applikationen er designet skalerbart, således brugeren ikke er begrænset til kun at kunne tilføje, fjerne eller redigere fra udelukkende en node, men derimod synkront holde databasen opdateret fra flere noder på en gang. Dette betyder at databasen og alle opkoblede administrationssystemer, altid vil være helt synkroniseret.

Design og struktur

Backend er designet på baggrund af MVVM designmønstret [3].



Figur 3.1: Oversigt over alle namespaces i administrationssystemet

På figur 3.1 ses det hvordan strukturen i administrationssystemet er opbygget.

- **Views** indeholder XAML og codebehindfilerne for den grafiske brugerflade.
- **ViewModels** indeholder de data og commands som views binder til.
- **Models** indeholder datamodeller og businesslogik.
- **Datamodels** indeholder datamodeller, herunder bl.a. produkter, collections og kategorier.
- **Events** indeholder events der bruges til kommunikation mellem viewmodel, den tilhørende aggregator samt parametremodellen til nogen af disse events.
- **SocketEvents** indeholder de events der bliver raised af Client ved modtagelse af data.
- **Brains** indeholder generel businesslogik (???) til bl.a. at oprette kommandoer til klienten, og fejl-håndtering.
- **Dependencies** indeholder sourcekode til at arbejde med commands.

- **Communication** indeholder klient til at skabe forbindelse til Central Server.

De ovenstående

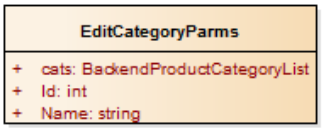
Presentation Layer
Business Logic Layer
ViewModels ligger som en mellemting mellem business logic layer og presentation layer. Der foregår en binding til fra viewsne til dataen i viewmodels.

Viewmodel kommunikation
Alle view startes fra MainViewViewModel, som invidviduelt starter deres egen viewmodel. Det betyder at det ville være nødvendigt at lave data globalt for at kunne arbejde på den samme data, som f.eks. kategorier og de individuelle produkter i disse.
For at løse dette, har vi valgt at bruge Prism EventAggregator [1], som tillader os at kommu-nikere på tværs af viewmodels. Når MainViewViewModel startes, subscriber den på en række forskellige events, som bliver rasied når følgende vinduer bliver loaded:

- AddProductWindow
- AddCategoryWindow
- EditProductWindow
- EditCategoryWindow
- DeleteCategoryWindow

MainWindowViewModel har da en metode for hver af disse events, som bliver kaldt når disse events bliver rasied. Når dette sker publisher MainWindowViewModel så et event, svarende til det vindue der er blevet loaded. Eksempelvis vil AddCategory's eventhandler publishe et event som AddCate-goryWindow subscriber på, med alle de nuværende kategorier som en parameter.

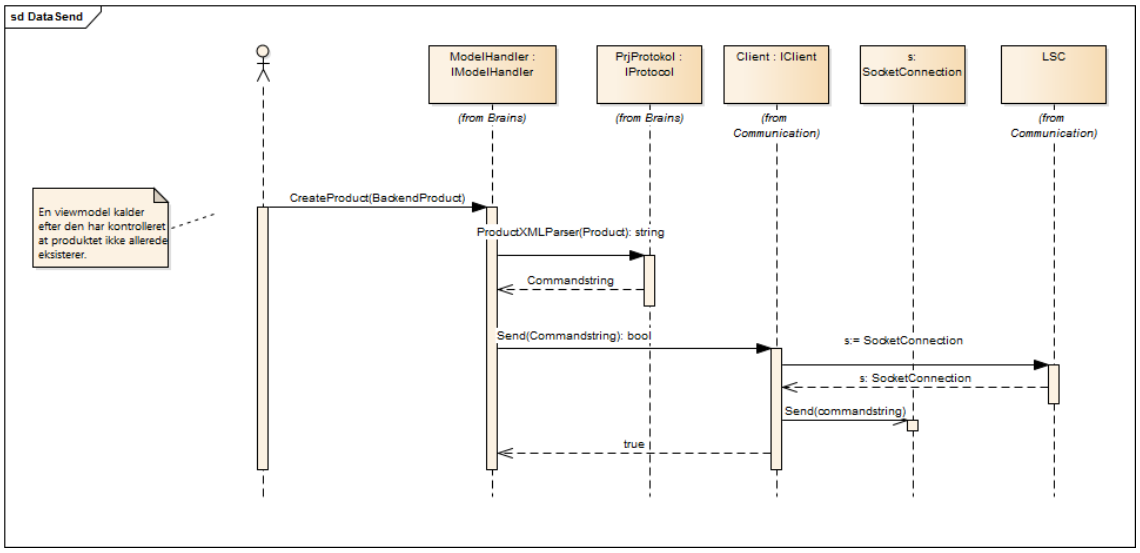
På denne måde undgår vi da, at der skal sendes data ind i et view, og videre ned i en viewmo-del. Hvis der er brug for mere end en parameter, eksempelvis EditProduct som skal bruge fire parametre, er det nødvendigt at lave sin egen parameter-type for dette (Backend.Models.Events : Parameters).



Figur 3.2: Eksempel på parametermodel, der i dette tilfælde er til når EditProductViewet bliver loaded

Data Access Layer
Kommunikation med Central Server Klienten i Administrationssystemet består af en sock-etconnection som er defineret i SharedLib[Ref]. Da denne skal bruges i både businesslogic, men også i MainWindowViewModel og der samtidig øn-skes at den samme benyttes hver gang, bruges Singleton[Reference] for denne forbindelse.
Til at sende data benyttes en ModelHandler [Ref], som hver i sær har en metode til de forskellige handlinger. Eksempelvis EditProduct, DeleteProduct mf. Disse metoder arbejder alle sammen på samme måde:

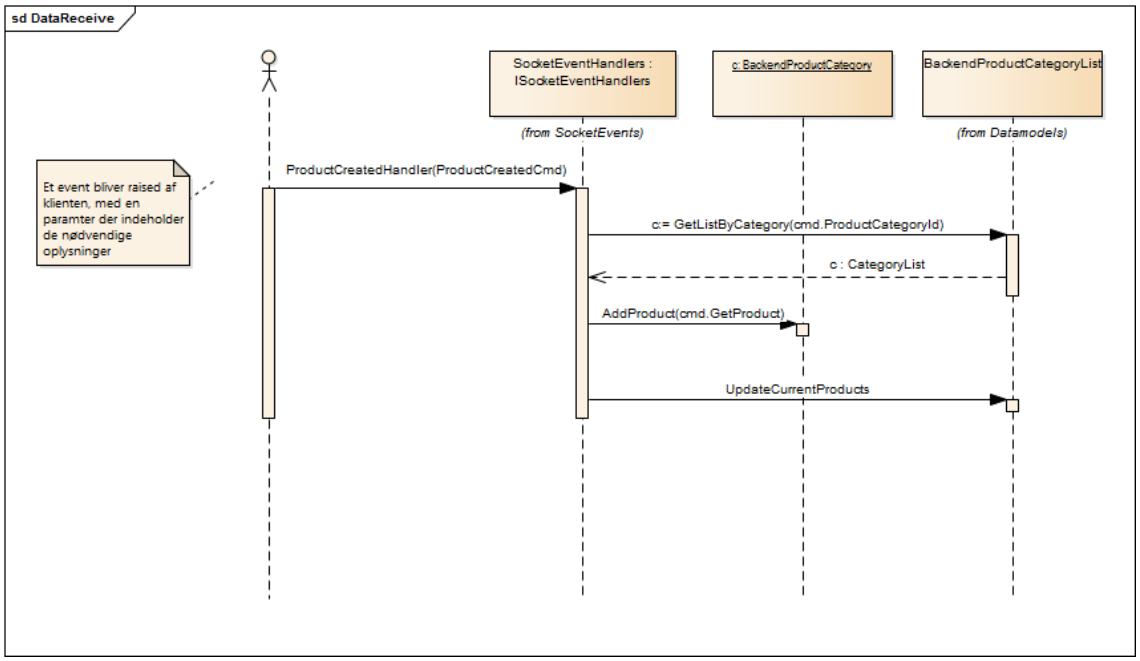
1. Opret en XML-kommandostring vha. protokollen [**REFERENCE TIL PROTOKOL-LEN**]
2. Send data til klienten



Figur 3.3: Eksempel på hvordan CreateProduct bliver behandlet i systemet og sendt til Central Server

Derefter afsluttes der. Det betyder at når der eksempelvis oprettes et nyt produkt, bliver der ikke opdateret noget i systemet, der bliver udelukkende sendt data til serveren.

For at modtage data, subscriber MainWindowViewModel på nogle forskellige events, som eksempelvis OnProductCategoryDeleted, OnProductEdited mf. Dette gøres igennem klassen SocketEventHandlers [Ref], som også indeholder de eventhandlere der bliver kaldt, når et event raises på serveren. Det er derfor disse eventhandlere som håndtere dataen, og lægger den nye data ind i datamodellerne – og det er først der, der bliver opdateret lokalt. På den måde vil der aldrig eksistere noget i databasen der ikke eksisterer i Administrationssystemet og vice versa.



Figur 3.4: Eksempel på hvordan produktet igen bliver modtaget fra serveren, efter det er blevet oprettet.

3.3 Central Server

3.4 Shared Library

4 Accepttest

Accepttest for UC1: Gennemfør salg

Use case under test: 1 - Gennemfør salg				
Scenarie: Hovedscenarie				
Prækondition: Kassesystemet har forbindelse til CentralServer, hvor i der eksisterer en Testvare 1 til 100 kr. samt en Testvare 2 til 50 kr				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Ekspedient trykker på "Testvare 1" til 100 kr	Testvare 1 fremkommer i listen over indtastede varer og den samlede pris for købet opdateres til 100 kr		
2	Ekspedient trykker på "Testvare 2" til 50 kr	Testvare 2 fremkommer i listen over indtastede varer og den samlede pris for købet opdateres til 150 kr		
3	Ekspedient indtaster 200	Beløbet fremkommer i GUI		
4	Ekspedient trykker på "Kontant Betaling"	Beløbet, som Kunde skal have retur, (50 kr.) fremkommer. Alle varer fjernes fra listen over indtastede varer. Samlet pris for varerne nulstillet til 0 kr. Salget gemmes i databasen		

Tabel 4.1: Accepttest 1: Gennemfør salg

Use case under test: 1 - Gennemfør salg				
Scenarie: Ext 1 - Forkert vare indtastet				
Prækondition: Der er ingen varer indtastet				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Ekspedient trykker på "Testvare 1" til 100 kr.	Testvare 1 fremkommer i listen over indtastede varer og den samlede pris for købet opdateres til 100 kr.		
2	Ekspedient vælger varen på listen over vare	Den valgte vare markeres		
3	Ekspedient trykker på slet knappen	Testvare 1 forsvinder fra listen over indtastede varer og den samlede pris for varerne nulstillet til 0 kr.		

Tabel 4.2: Accepttest 1: Gennemfør salg - Ext 1

Use case under test: 1 - Gennemfør salg				
Scenarie: Ext 2 - Forkert antal indtastet				
Prækondition: Der er ingen varer indtastet				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Ekspedient trykker på tallet 2	Tallet 2 fremkommer i displayet		
2	Ekspedient trykker på "Testvare 1" til 100 kr.	Testvare 1 fremkommer i listen over indtastede varer gange 2 og den samlede pris for købet opdateres til 200 kr.		
3	Ekspedient vælger varen på listen over vare	Den valgte vare markeres		
4	Ekspedient indtaster 5	Tallet 5 fremkommer i displayet		
5	Ekspedient trykker på "Antal"	I listen over indtastede varer står der nu 5 i antal ved Testvare 1. Samlet pris ændres til 500 kr		

Tabel 4.3: Accepttest 1: Gennemfør salg - Ext 2

Use case under test: 1 - Gennemfør salg				
Scenarie: Ext 3 - Kunden stopper handlen				
Prækondition: Der er ingen varer indtastet				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Ekspedient trykker på "Testvare 1" til 100 kr.	Testvare 1 fremkommer i listen over indtastede varer og den samlede pris for købet opdateres til 100 kr.		
2	Ekspedient trykker på "Annuller"	Alle varer fjernes fra listen over indtastede varer. Samlet pris for varerne nulstillet til 0 kr.		

Tabel 4.4: Accepttest 1: Gennemfør salg - Ext 3

Accepttest for UC2: Returner vare

Use case under test: 2 - Returner vare				
Scenarie: Hovedscenarie				
Prækondition: Kassesystemet har forbindelse til Central Server, hvor i varen der eksisterer en Testvare 1 til 100kr				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Ekspedient trykker på "Testvare 1" til 100 kr	Testvare 1 fremkommer i listen over indtastede varer og den samlede pris for købet opdateres til 100 kr		
2	Ekspedient trykker på testvaren på Salgs listen	Testvaren markeres		
3	Ekspedient indtaster 1 via tastatur i GUI	1 fremkommer i display		
4	Ekspedient vælger Retur	Testvare i salgsliste sættes til antal -1		
5	Ekspedient indtaster 0 + kontant	Gui angiver at der skal returneres 100 kr		

Tabel 4.5: Accepttest 2: Returner vare

Use case under test: 2 - Returner vare				
Scenarie: Ext 1 - Kunde fortryder				
Prækondition: Der er varer indtastet				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Ekspedient trykker Annuller	Salgsliste rydes		

Tabel 4.6: Accepttest 2: Returner vare - Ext 1

Accepttest for UC3: Opret Produkt

Use case under test: 3 - Opret Produkt				
Scenarie: Hovedscenarie				
Prækondition: Der er oprettet forbindelse til CentralServer, Produktet "TestProdukt 1"eksisterer *ikke* i systemet, samt *Testkategori 1* kategorien eksisterer i systemet.				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger trykker på "Opret Produkt" i Administrationssystem GUI	Et vindue med Produkt formular fremkommer		
2	Supahburger indtaster Navn: "TestProdukt 1", Pris: "100", Produktkategori: "Testkategori 1"og trykker på knappen til Gem	En beskesboks informerer om at data er sendt til serveren, og vinduet til opret Produkt lukkes.		
3	Supahburger observerer Produktlisten	TestProdukt 1 kommer frem i Produktlisten.		

Tabel 4.7: Accepttest 3: Opret Produkt

Use case under test: 3 - Opret Produkt				
Scenarie: Extension 1: Produktet eksisterer allerede				
Prækondition: Der er oprettet forbindelse til CentralServer, Produktet "TestProdukt 1"eksisterer *allerede* i systemet, samt "Testkategori 1"kategorien eksisterer i systemet.				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger trykker på "Opret Produkt" i Administrationssystem GUI	Et vindue med Produkt formular fremkommer		
2	Supahburger indtaster Navn: "TestProdukt 1", Pris: "100", Produktkategori: "Testkategori 1"og trykker på knappen til Gem	En beskesboks informerer Supahburgere om at Produktet allerede eksisterer, og vinduet til opret Produkt forbliver.		

Tabel 4.8: Accepttest 3: Opret Produkt, extension 1.

Use case under test: 3 - Opret Produkt				
Scenarie: Extension 2: CentralServer melder fejl				
Prækondition: Der er oprettet forbindelse til CentralServer, der er *ikke* forbindelse til database.				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger trykker på “Opret Produkt” i Administrationssystem GUI	Et vindue med Produkt formular fremkommer		
2	Supahburger indtaster Navn: "TestProdukt 1", Pris: "100", Produktkategori: "Testkategori 1"og trykker på knappen til Gem	En beskesboks informerer Supahburger om at der er sket en fejl, vinduet til opret Produkt lukker.		

Tabel 4.9: Accepttest 3: Opret Produkt

Accepttest for UC4: Rediger Produkt

Use case under test: 4 - Rediger Produkt				
Scenarie: Hovedscenarie				
Prækondition: Administrationssystem har forbindelse til centralserver, og der er et testProdukt med navn "TestProdukt 1"og pris "100"samt er i kategorien "Testkategori 1"og "Testkategori 2"i systemet.				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger vælger "TestProdukt 1"	TestProdukt 1 markeres i Produktlisten		
2	Supahburger trykker på knappen til rediger Produkt	Et nyt vindue præsenteres for Supahburger, som viser hhv. pris, navn og nuværende Produktkategori for det markerede "Testprodukt 1"		
3	Supahburger ændrer navnet til TestProdukt 1a, prisen til 150 samt Produktkategori til Produktkategori 2, hvorefter han trykker på knappen til gem.	Supahburger bliver præsenteret for en besked, der fortæller at data er sendt til serveren, og "Rediger Produkt-vinduet bliver lukket		
4	Supahburger observerer Produktlisten.	TestProdukt 1 ændres til TestProdukt 2, prisen til 150 og kategorien til Testkategori 2, automatisk		

Tabel 4.10: Accepttest 4: Rediger Produkt

Use case under test: 4 - Rediger Produkt				
Scenarie: Ext. 1 - Supahburger vælger at annullere				
Prækondition: Der skal være forbindelse til centralserver, "TestProdukt 1"er oprettet samt "Testkategori 1"eksisterer.				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger vælger "TestProdukt 1"	TestProdukt 1 markeres i Produktlisten		
2	Supahburger trykker på knappen til rediger Produkt	Et nyt vindue præsenteres for Supahburger, som viser hhv. pris, navn og nuværende Produktkategori for det markerede "Testprodukt 1"		
3	Supahburger trykker på knappen til Annuller	Vinduet til rediger Produkt lukkes		

Tabel 4.11: Accepttest 4: Rediger Produkt, ext. 1

Use case under test: 4 - Rediger Produkt				
Scenarie: Ext. 2 - Central server melder fejl				
Prækondition: Administrationssystem har forbindelse til centralserver, databasen er *ikke tilgængelig*. Der er et testProdukt med navn "TestProdukt 1"og pris "100"samt er i kategorien "Testkategori 1"og "Testkategori 2"i systemet.				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger vælger "TestProdukt 1"	TestProdukt 1 markeres i Produktlisten		
2	Supahburger trykker på knappen til rediger Produkt	Et nyt vindue præsenteres for Supahburger, som viser hhv. pris, navn og nuværende kategori for det markerede "Testpodukt 1"		
3	Supahburger ændrer navnet til TestProdukt 1a, prisen til 150 samt kategori til Kategori 2, hvorefter han trykker på knappen til gem.	Supahburger bliver præsenteret for en besked, der fortæller at der skete en fejl, og "Rediger Produkt-vinduet bliver lukket		

Tabel 4.12: Accepttest 4: Rediger Produkt ext. 2

Accepttest for UC5: Slet produkt

Use case under test: 5 - Slet produkt				
Scenarie: Hovedscenarie				
Prækondition: Der er oprettet et "TestProdukt 1", som er vist på listen over produkter				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger vælger "TestProdukt 1"på listen over produkter og trykker slet	Det valgte produkt er valgt på listen og et vindue åbner, som spørger om Supahburger virkelig ønsker at slette produktet.		
2	Supahburger trykker på knappen til at godkende sletningen	Vinduet lukker igen og en meddelelse om, at produktet er slettet.		
3	Det valgte produkt slettes umiddelbart efter fra listen	Produktet forsvinder fra listen over produkter.		

Tabel 4.13: Accepttest 5: Slet produkt

Use case under test: 5 - Slet produkt				
Scenarie: Ext 1 - Supahburger vælger at annullere sletningen.				
Prækondition: Der er oprettet et "TestProdukt 1", som er vist på listen over produkter				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger vælger "TestProdukt 1"på listen over produkter og trykker slet	Det valgte produkt er valgt på listen og et vindue åbner, som spørger om Supahburger virkelig ønsker at slette produktet.		
2	Supahburger trykker på knappen til at annullere sletningen	Vinduet lukker igen og intet yderligere foretages.		

Tabel 4.14: Accepttest 5: Slet produkt - Ext 1

Use case under test: 5 - Slet produkt				
Scenarie: Ext 2 - Central server melder fejl				
Prækondition: Der er oprettet et produkt, som er vist på listen over produkter				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger vælger "TestProdukt 1"på listen over produkter og trykker slet	Det valgte produkt er valgt på listen og et vindue åbner, som spørger om Supahburger virkelig ønsker at slette produktet.		
2	Supahburger trykker på knappen til at godkende sletningen	Vinduet lukker igen og en meddelelse om, at produktet er slettet.		
3	Det valgte produkt opdateres ikke og der vises en fejlbesked	Produktet bliver på listen over produkter, og en fejlbesked vises til Supahburger.		

Tabel 4.15: Accepttest 5: Slet produkt - Ext 2

Accepttest for UC6: Opret produktkategori

Use case under test: 6 - Opret produktkategori				
Scenarie: Hovedscenarie				
Prækondition: Administrationssystemet har oprettet forbindelse til server, "Testprodukt 1"eksisterer *ikke* i systemet. samt "Testkategori 1"eksisterer.				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger trykker på knappen til Opret produkt	Supahburger præsenteres for et nyt vindue		
2	Supahburger indtaster Navn: "Testprodukt 1", Pris: "100"og Kategori: "Testkategori 1, hvorefter Supahburger trykker på gem.	Supahburger bliver præsenteret for en beskedboks der informerer om at produktet er gemt, og vinduet til opret produkt lukkes.		
3	Supahburger observerer produktlisten	Testprodukt 1 vises i produktlisten		

Tabel 4.16: Accepttest 6: Opret produktkategori

Use case under test: 6 - Opret produktkategori				
Scenarie: Extension 1: Kategorien eksisterer allerede				
Prækondition: Administrationssystem har oprettet forbindelse til CentralServer, "Testprodukt 1"eksisterer *allerede* i systemet, samt "Testkategori 1"eksisterer.				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger trykker på knappen til Opret produkt	Supahburger præsenteres for et nyt vindue		
2	Supahburger indtaster Navn: "Testprodukt 1", Pris: "100"og Katetegori: "Testkategori 1, hvorefter Supahburger trykker på gem.	Supahburger bliver præsenteret for en beskedboks der informerer om at produktet allerede eksisterer, og vinduet til opret produkt forbliver.		

Tabel 4.17: Accepttest 6: Opret produktkategori, ext. 1

Use case under test: 6 - Opret produktkategori				
Scenarie: Extension 2: Der er oprettet forbindelse til CentralServer, der er *ikke* forbindelse til databasen				
Prækondition: Administrationssystem har *ikke* oprettet forbindelse til CentralServer, "Testprodukt 1"eksisterer *ikke* i systemet, samt "Testkategori 1"eksisterer.				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger trykker på knappen til Opret produkt	Supahburger præsenteres for et nyt vindue		
2	Supahburger indtaster Navn: "Testprodukt 1", Pris: "100"og Kategori: "Testkategori 1, hvorefter Supahburger trykker på gem.	Supahburger bliver præsenteret for en beskedboks der informerer om at der ikke er forbindelse til CentralServer, og vinduet til opret produkt lukkes.		

Tabel 4.18: Accepttest 6: Opret produktkategori, ext. 2

Accepttest for UC7: Rediger produktkategori

Use case under test: 7 - Rediger produktkategori				
Scenarie: Hovedscenarie				
Prækondition: Forbindelse til CentralServer er oprettet. Der er oprettet en "Kategori 1".				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger trykker på kategori "Kategori 1"på listen over kategorier	"Kategori 1"er er markeret på listen over kategorier.		
2	Supahburger trykker på knappen til at redigere produkt	Et vindue åbner med kategoriens nuværende navn		
3	Supahburger redigerer navnet på kategorien til "Kategori 123"og trykker på knappen til at gemme ændringen	Vinduet lukker og der bliver vist en besked til Supahburger der bekræfter, at ændringen er foretaget.		
4	Supahburger observerer kategorilisten, for at se, at ændringen er foretaget	"Kategori 1"skifter navn til "Kategori 123"		

Tabel 4.19: Accepttest 7: Rediger produktkategori

Use case under test: 7 - Rediger produktkategori				
Scenarie: Ext 1: CentralServer melder fejl				
Prækondition: Forbindelse til CentralServer er *ikke* oprettet. Der er oprettet en "Kategori 1".				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger trykker på kategori "Kategori 1"på listen over kategorier	"Kategori 1"er markeret på listen over kategorier.		
2	Supahburger trykker på knappen til at redigere produkt	Et vindue åbner med kategoriens nuværende navn		
3	Supahburger redigerer navnet på kategorien til "Kategori 123"og trykker på knappen til at gemme ændringen	Vinduet lukker.		
4	Supahburger observerer kategorilisten	Kategorien fjernes *ikke* fra listen over kategorier, og der vises en fejlbesked til Supahburger.		

Tabel 4.20: Accepttest 7: Rediger produktkategori - Ext 1

Accepttest for UC8: Slet produktkategori

Use Case under test: 8 - Slet Produktkategori				
Scenarie: Hovedscenarie				
Prækondition: Der er forbindelse til , Testkategori 1 og Testkategori 2 eksisterer i systemet og Testprodukt 1 er oprettet og placeret i Testkategori 1				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger vælger Testkategori 1 i kategorilisten	Testkategori 1 markeres.		
2	Supahburger trykker på knappen til Slet Kategori	En dialogboks åbnes der informerer Supahburger om at der er produkter i kategorien, og Supahburger kan vælge en anden kategori de skal flyttes til.		
3 4 5	Supahburger trykker vælger Testkategori 2 og trykker ok Supahburger observerer kategorilisten Supahburger vælger Testkategori 2 i kategorilisten og observerer produktlisten	Vinduet til slet produkter og information lukkes Testkategori 1 forsvinder fra listen produktlisten indeholder Testprodukt 1		

Tabel 4.21: Accepttest 8: Slet produktkategori

Use case under test: 8 - Slet produktkategori				
Scenarie: Ext. 2 Supahburger trykker annuller				
Prækondition: Der er forbindelse til CentralServer, Testkategori 1 eksisterer, Testprodukt 1 er oprettet og placeret i Testkategori 1				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger vælger Testkategori 1 i kategorilisten	Testkategori 1 markeres.		
2	Supahburger trykker på knappen til Slet Kategori	En dialogboks åbnes der informerer superbruger om at der er produkter i kategorien, og Supahburger kan vælge en anden kategori de skal flyttes til.		
3 4 5	Supahburger trykker på knappen til Anuller Supahburger observerer kategorilisten Supahburger vælger Testkategori 2 i kategorilisten og observerer produktlisten	Vinduet til slet produkter og information lukkes Testkategori 1 forsvinder fra listen af produkter produktlisten indeholder Testprodukt 1		

Tabel 4.22: Accepttest 8: Slet produktkategori, extension 1

Use case under test: 8 - Slet produktkategori				
Scenarie: Ext. 1 Der er ingen produkter i kategorien				
Prækondition: Der er forbindelse til CentralServer, Testkategori 1 eksisterer og er tom				
Step	Handling	Forventet resultat	Observation	Vurdering (OK/fejl)
1	Supahburger vælger Testkategori 1 i kategorilisten	Testkategori 1 markeres.		
2	Supahburger trykker på knappen til Slet Kategori	En dialogboks åbnes hvori der står at der ingen produkter er.		
3 4 5	Supahburger trykker på knappen til Slet Supahburger observerer kategorilisten Supahburger vælger Testkategori 2 i kategorilisten og observerer produktlisten	Vinduet til slet produkter og information lukkes Testkategori 1 forsvinder fra listen produktlisten indeholder Testprodukt 1		

Tabel 4.23: Accepttest 8: Slet produktkategori, extension 2

Glossary

Supahburger	En bruger med yderligere rettigheder end Ekspedient, som kan administrere systemets indhold i Administrationssystemet
Ekspedient	En bruger der benytter Kassesystem til at saelge varer
Kunde	En person, som koeber eller returner varer
Kasseapparat	En skaerm, med en grafisk brugerflade, hvorpaa Ekspedient haandterer salg
Administrationssystem	Der hvor Superbruger administrerer produkter og produktkategorier
CentralServer	Er bindeleddet imellem Kasseapparat, Administrationssystem og Databasen
Database	Den centrale database der administreres af CentralServer
SharedLib	Et bibliotek til at encode/decode xml, samt indeholder Produkt og ProduktKategori klasserne
GUI	Grafisk brugergraenseflade
Produkt	En vare, som en kunde kan koebe og returnere
Produktkategori	En gruppe af produkter

Tabel 4.24: Glossary

Litteratur

[1] Prism event aggregator. https://msdn.microsoft.com/en-us/library/ff921122.aspx. Tilgået: 08/12-2015.

[2] Tcp/ip protocol architecture. https://technet.microsoft.com/en-us/library/cc958821.aspx. Tilgået: 08/12-2015.

[3] Poul Ejner Rovsing. The model-view-viewmodel design pattern. pages 5–36, 2014.