CDIO 3-ugers, Juni 2015

02324 Videregående programmering

Projektnavn: 16_CDIO_final
Emne: MediCom Softwaresystem 2.0
Afleveringsfrist: Fredag d. 19/06-2015, kl. 12:00
CDIO gruppenummer: 16

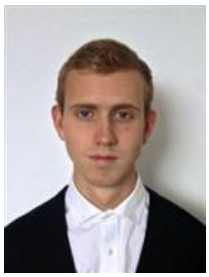
Gruppemedlemmer (studienummer, efternavn, fornavn):

s073629, Jabbari, Muhammad Bilal

s123157, Adamsen, Frederik

s144846, Eriksen, Robert

s144845, Ørnby, Victor



s144844, Thomsen, Mads

s144858, Hansen, Nicklas

Rapporten er afleveret elektronisk via Campusnet og der skrives derfor ikke under. Rapporten indeholder 60 sider inkl. denne side og bilag.

Code repository URL: https://github.com/nicklas669/16_CDIOFinal

Timeregnskab

		Opgaver					
Dato	Deltager	Analyse og design	Implementation	Test	Dokumentation	Andet	I alt
03-06-2015	Bilal	4					4
04-06-2015	Bilal	3			1	1	5
06-06-2015	Bilal					1	1
08-06-2015	Bilal		4				4
09-06-2015	Bilal		4			1	5
10-06-2015	Bilal		3	2		1	6
11-06-2015	Bilal		3	1		1	5
14-06-2015	Bilal					1	1
15-06-2015	Bilal	2		2			4
16-06-2015	Bilal			2		2	4
17-06-2015	Bilal	1		2		2	5
18-06-2015	Bilal			2	1	1	4
04-06-2015	Frederik		8				8
08-06-2015	Frederik		4		1		5
09-06-2015	Frederik		4				4
10-06-2015	Frederik	1	4		1		6
14-06-2015	Frederik	1		3	1		5
15-06-2015	Frederik			2			2
17-06-2015	Frederik			5			5
18-06-2015	Frederik				3		3
03-06-2015	Mads	2				2	4
04-06-2015	Mads	2					2
08-06-2015	Mads	1	3		1		5
09-06-2015	Mads		4			1	5
14-06-2015	Mads				2	1	3
15-06-2015	Mads				3	1	4
16-06-2015	Mads				4		4
17-06-2015	Mads				5		5
18-06-2015	Mads				4		4
04-06-2015	Nicklas	4	2				6
06-06-2015	Nicklas		2			1	3
07-06-2015	Nicklas		1		1	1	3
08-06-2015	Nicklas		4		1	1	6
10-06-2015	Nicklas	1	4	1	1		7

11-06-2015	Nicklas		4	2	1		7
13-06-2015	Nicklas		2	3			5
14-06-2015	Nicklas		1	1	4		6
15-06-2015	Nicklas				4	1	5
16-06-2015	Nicklas				5	1	6
17-06-2015	Nicklas				5	1	6
18-06-2015	Nicklas				5	1	6
04-06-2015	Robert	2			1		3
06-06-2015	Robert		3				3
08-06-2015	Robert		5				5
09-06-2015	Robert		5				5
10-06-2015	Robert		4		2		6
11-06-2015	Robert		4		1		5
12-06-2015	Robert		4		1		5
14-06-2015	Robert				2		2
15-06-2015	Robert			1	3		4
18-06-2015	Robert				6		6
04-06-2015	Victor	4					4
05-06-2015	Victor	1.5					1.5
06-06-2015	Victor	1.5					1.5
07-06-2015	Victor				1	1	2
08-06-2015	Victor		4				4
10-06-2015	Victor	1	4				5
13-06-2015	Victor		2		2	1	5
14-06-2015	Victor				2		2
17-06-2015	Victor				2		2
18-06-2015	Victor				4		4
09-06-2015	Victor		5				5

Brugsvejledning for systemet

Det udviklede system indeholder mange komponenter og det kan således være vanskeligt at finde rundt i. Følgende afsnit er ment som en kort oversigt over komponenterne i systemet, samt hvordan de startes op - disse kan ses forneden.

Databasen

Datalaget for det samlede system udgøres af en MySQL database. I projektet bruges en online database som vi har fået udleveret til formålet. Databasen kan også køres lokalt idet der i projektet er vedlagt et SQL-script med et dump af databasen, som kan bruges til at lave en "kopi" af databasen.

GWT (Webadministration)

Før GWT webapplikationen startes, skal der køre en MySQL database server i baggrunden. Denne database server kan enten køre lokalt eller ude på netværket. Som standard er vores GWT applikation koblet sammen med den MySQL server, vi har fået udleveret i løbet af dette projekt. Dette betyder, at computeren, der kører GWT applikationen, skal være forbundet til internettet før applikationen startes. Hvis der ønskes at forbinde til en anden database end den der er valgt som standard, skal dette ændres i String variablen "URL" i DAO-klassen i pakken "dtu.server.dal". GWT applikationen kan herefter startes.

Bruger-id og passwords

Når GWT applikationen er oppe at køre og man er inde på hjemmesiden, skal der logges ind for at komme videre i systemet. Dette kan gøres med følgende bruger id og kodeord:

- Superbruger login: ID: 1, password: passTE01.
- Farmaceut login: ID: 2, password: passJH02.
- Værkfører login: ID: 3, password: passGJ03.
- Operatør login: ID: 4, password: passJS04.

Det skal dog nævnes, at der i forbindelse med brugervenlighed til test af systemet, er oprettet knapper på forsiden der kan bruges til at logge ind direkte som hver af de aktørtyper der findes i systemet.
GWT applikationen startes i vores projekt ved at højreklikke på pakken "GWT" og klikke "Run as" og derefter "Web Application (GWT Super Dev Mode)".

ASE (Afvejning Styrings Enhed)

Inden ASE startes, skal det vælges om den skal forbinde til vores vægtsimulator eller en rigtig, fysisk Mettler Toledo vægt. Dette gøres inde i kildekoden ved at ændre fieldet “realScale” til true eller false. Dette er nødvendigt fordi der er små forskelle mellem vores vægtsimulator og den rigtige vægt.

Inden ASE startes, er der flere ting der skal køre i baggrunden. Der skal køre den førnævnte MySQL database server men derudover skal der også køre en vægt. Vægten kan være vores egen vægtsimulator eller en rigtig, fysisk Mettler Toledo vægt. Når disse to ting er oppe at køre og host-maskinen er forbundet til dem via netværket, kan ASE startes og en afvejning kan finde sted.

ASE startes i vores projekt ved at højreklikke på pakken “ASE” og klikke “Run as” og derefter “Java Application”.

Indholdsfortegnelse

Indledning

Projektet er sidste del af et gennemgående CDIO system der omhandler et firma, med en netværksvægt. Man skulle kunne styre en vægt over netværket og registrere ting i en database. Undervejs i dette semester har vi i flere forskellige kurser haft mindre projekter inden for dette emne og i dette "final" projekt skal alle disse mindre dele ikke længere stå alene men bringes sammen og arbejde sammen.

Dette projekt, også kendt under navnet CDIO final, udmærker sig ved, på den ene side at være en øvelse i at samle tidligere applikationer til ét stort system, samtidigt med at udbygge og forbedre funktionaliteten for disse tidligere applikationer.

Der er rigtig mange forbedringer og funktioner man kunne foretage sig i projektets komponenter og det fordrer dermed en god projektstruktur og velovervejede tanker omkring omfanget af projektet og hvorvidt man må afgrænse sig. Det er en balance mellem hvor meget vi kan nå og den tid, projektperioden strækker sig over.

I projektperioden har vi gjort brug af udviklingsmetoden Unified Proces. Denne går ud på at gøre en iterativ og inkremental form for udvikling. Der er således ikke en rigtig måde at gøre tingene på fra start af, det er en iterativ proces hvor man starter et sted, arbejder lidt på det, bliver klogere, arbejder videre, tester, og sådan kører det i ring.

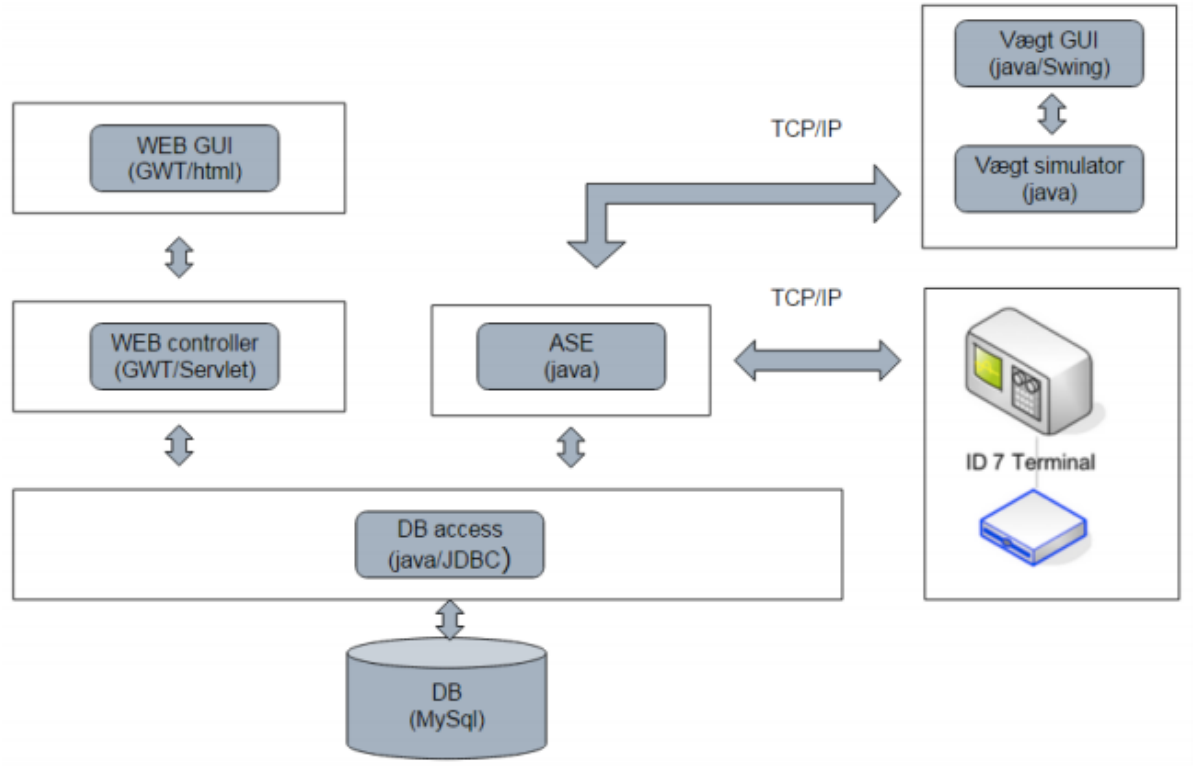
For at skabe klarhed af systemets arkitektur, vil vi i rapportens første del beskrive systemets arkitektur på tværs af applikationerne ved hjælp af et overordnet diagram.

Da CDIO final bygger på tidligere projekter vil vi kort beskrive disse applikationer. De tidligere udviklede applikationer har undervejs i 3-ugers perioden undergået store forandringer for at kunne fungere i sammenspil med de andre dele samt for at opfylde de nye krav til systemet.

Til sidst vil vi gennemgå GWT applikationen fra et analyse til design perspektiv. Vi vil i den forbindelse vedhæfte brugervejledninger til hvordan man efterprøver hele systemet. Hele rapporten vil i det omfang, det er relevant, inddrage diagrammer og modeller for at understøtte samt dokumentere applikationernes funktionalitet.

Det overordnede CDIO system

Forneden, på figur 1, ses et diagram over systemets overordnede struktur. Diagrammet er et udsnit taget fra oplægget til dette projekt og efter diagrammet vil det kort gennemgås hvad diagrammet viser.



Figur 1: Overordnet CDIO system

Af diagrammet for oven kan man se de forskellige dele der udgør hele afvejningssystemet. WEB GUI'en i venstre hjørne er den grafiske brugerflade hvormed Administrator, Farmaceut og Værkfører tilgår systemet fra. Disse har hver især forskellige beføjelser til at tilgå denne applikation, som vil blive gennemgået senere i rapporten. Under 3-ugers perioden er det især Web GUI'en der er blevet udbygget med ny funktionalitet i forhold til den tidligere iteration.

I bunden ses database-laget der skal implementeres som en MySQL database. Lige omkring midten ses ASE (afvejnings styrings enhed) der skal fungere som led mellem operatør, der står ved en vægt, og datalaget.
Endeligt, ude til højre, ses netværksvægten - øverst som en simulator og neder som den reelle fysisk Mettler Toledo netværksvægt.

Oprettelse af produktbatch

For at illustrere hvordan systemet fungerer vil vi tage udgangspunkt i et par use cases der tilsammen involverer alle dele af systemet. Den første use case er en "værkfører" der ved hjælp af GUI'en opretter et produktbatch (f.eks. at der skal produceres "Saltvand m. citron"). I forlængelse af den usecase vil vi beskrive hvordan en operatør bruger dette nyoprettede produktbatch til afvejning af de nødvendige raavarer.

- Det første der sker er, at Værkfører logger på Web applikationen. Hans id og password bliver tjekket i databasen.
- Nu trykker han "Opret produktbatch" og indskriver:
    - Produktbatch ID som han selv finder på.
    - Status - 0 fordi produktionen af produktbatch endnu ikke er på påbegyndt.
    - Recept ID - det er et ID der skal findes i forvejen.
- Herefter trykkes "Tilføj produktbatch" og produktbatchen oprettes i databasen, forudsat at der er givet et gyldigt produktbatch ID og recept ID. Herefter kan en operatør via ASE og en vægt producere en batch af den givne recept.

Ovenstående Usecase illustrerer ved oprettelsen af produktbatch sammenspillet mellem Web Applikationen og Databasen. GWT Servlet er den del del der reelt kommunikerer med databasen ved DAO og DTO men dette vil blive gennemgået nærmere senere i rapporten.

Operatør bruger vægten

For at illustrere alle dele af systemet er det relevant at inddrage en sidste usecase der går i forlængelse af værkfører produktbatch scenariet. En operatør bruger en udskrevet Produktions forskrift der indeholder (produktbatch ID). Operatøren gør derefter:

- Indskriver operatør ID på vægten og for adgang.
- Derefter indtaster han det produktbatch id der er oplyst på produktions forskriften.
- Vægten prompter operatøren for recept navn.
- Derefter bliver vægten automatisk tareret af ASE'en.
- Operatøren skal derefter sætte beholder på vægten.
- Vægten beder operatøren om at afveje en råvare i en bestemt mængde.
- Dette forsætter indtil alle råvarer er blevet afvejet korrekt.

Begge Usecases illustrerer hvordan web applikationen og vægten kommunikerer gennem databasen. ASE står for at videregive og opdaterer informationen fra vægten til databasen. I løbet af rapporten vil vi give et mere detaljeret indblik hvordan disse applikationer mere konkret kommunikere sammen.

Projekt Plan

Grundige tanker omkring planlægning og fordelingen af arbejdsopgaver er en essentiel del i at få et velfungerende CDIO projekt. Tidligt i processen lavede vi en prioriteret liste over hvilke funktioner der var vigtigst at få implementeret.

Efter at have testet systemet og konstateret at flere ældre CDIO projekter, heriblandt ASE'en og vægt simulatoren, ikke fungerede optimalt, opstod der en naturlig opdeling af arbejdsopgaver. Groft sagt bestod gruppen af en "GWT", en "ASE" og en "Vægt simulator" opdeling, med naturlige overlap og vidensdeling. Dette gjorde at vi 7 dage inden afleveringsfristen havde muligheden for at teste flere usecases' med flere dele af systemet og her meget tidligt havde en fungerende prototype oppe at køre.

Afgrænsning

Ved større projekter er det vigtig at kunne dokumentere hvad man har opnået samt hvilke funktioner der ikke nåede deadline. Da projektet er et sammensurium af tidligere projekter har vi valgt at lave et afgrænsningsafsnit for hver komponent i systemet. På den måde bliver de manglende funktionalitet dokumenteret i forlængelse af det relevante afsnit. Selvom afgrænsning inddrager tidligere projekter vil hovedfokus i dette projekt være på GWT.

Import fra tidligere CDIO Projekter

CDIO projekterne følger en iterativ struktur hvilket gør at man kan bruge elementer fra tidligere projekter. Nedenfor vil vi kort gennemgå de dele vi i større eller mindre grad har genbrugt eller arbejdet videre på i udviklingen af dette projekt.

- Password validation fra CDIO1: https://github.com/nicklas669/16_del1

I CDIO 1 i dette kursus lavede vi validering af passwords. Dette har vi kunnet genbruge dele af til at lave password validering på vores tekstbokse.

- CDIO2: https://github.com/nicklas669/16_CDIO2/commits/master

I CDIO2 i dette kursus lavede vi en vægt simulator der skulle simulere en fysisk Mettler Tolede vægt. Denne har vi videreudviklet på i løbet af dette projekt.

- CDIO3: https://github.com/nicklas669/16_CDIO3_DB

I CDIO3 i dette kursus udviklede vi en GWT webapplikation der udgjorde et personkartotek. Dette projekt har vi brugt som base til vores GWT applikation i dette projekt og blot arbejdet videre på det.

- Eksamensprojekt fra Indledende databaser og database programmering: Github repository: https://github.com/RobertEriksen/DB

Vi har haft et eksamensprojekt i kurset Indledende database og databaseprogrammering, hvori vi lavede en database på 3. normalform som vi stort set kunne genbruge til dette projekt. Der skulle selvfølgelig foretages lidt småændringer men i det store hele har den database været en god inspirationskilde til den database vi bruger i dette projekt.

Vægt Simulator

Følgende afsnit vil kort beskrive hvordan vægt simulatoren er blevet ændret under CDIO final for at kunne fungere i sammenspil med de andre applikationer. Kapitlet bygger videre på vægtsimulatoren fra CDIO 2 i videregående programmering, og opridser konkret hvilke funktioner der er blevet ændret. Da vi allerede har afleveret dokumentation for vægt simulatoren i et andet projekt og siden at simulatoren blot er en lille del af dette projekt, vil vi ikke gentage os selv, men i stedet fokuserer på ændringerne.

RM20 kommando

Den omtalte netværksvægt overholder en vis protokol, altså en række kommandoer den forstår. En del af denne protokol var en såkaldt “RM20” kommando. Denne kommando kan man sende til vægten, det er dette ASE bruges til, og så beder vægten om input rent aktivt fra en operatør der står ved vægten og derefter svarer vægten tilbage til ASE med svaret.

I CDIO 2, hvor vi skulle udvikle denne simulator som skulle gøre det ud for den rigtige vægt, havde vi en fejl der knyttede sig til RM20 kommandoen. Programmet opførte sig som om at input-tråden(der har ansvar for at læse input fra konsollen) ikke lukkede korrekt ned. Dette skyldes at der var to processer der anvendte Scanner.in istedet for én. Der medførte ligeledes at der ved yderligere RM20 kommandoer blev åbnet nye tråde der alle scannede efter input fra konsollen.

Årsagen til problemet stammede fra af en misforståelse i forhold til hvordan at man korrekt lukker for en tråd i Java. Løsningen lå i kun at genstarte menu-tråden, der har ansvar for at simulere menuen på vægten, og ikke input tråden. Funktionslaget var skrevet således at den behandlede input fra konsollen uafhængigt af hvilken input tråd den modtager fra, så det var ikke nødvendigt at ændre i input-klassen.
Der var altså en del rod med tråde, men vi er nået frem til et resultat der er brugeligt.

Menu-klassen

I forbindelse med ændringen i menu-klassen opstod en ny fejl. Menu-klassen viste ikke korrekt det rigtige output i det primære display. Det var i stedet kun det sekundære display der blev opdateret.

Korrekt opførsel ville være, at det var det primære display, der viser den nuværende vægt på displayet. Årsagen var, at koden kontrollerede for hvorvidt main display eller sekundært display skulle opdateres, og hvis en af dem var opdateret ville koden skrive både det primære display og det sekundære. Dette var til trods for, at et af dem kunne risikerer at være tomt/blankt. Løsningen lå i at at kontrollere for hvilken af de to display der har ændret sig, og kun opdatere den af dem, det var relevant for.

Et problem der opstod efter ovenstående fejlrettelser var, at RM20 kommandoen ikke blev gennemført korrekt medmindre der på forhånd fra ASE var givet en tekststreng via P111 kommandoen der vises i det sekundære display. Vi har ikke rettet fejlen idet vi mener vi kan arbejde uden om den, men det er værd at notere at RM20 ikke fungerer korrekt medmindre der er en besked at vise i det sekundære display. Dette er vigtigt i forbindelse med udvikling af ASE'en.

ASE (Afvejnings Styrings Enhed)

ASE'en der kan ses i midten på det overordnede system arkitektur diagram, figur 1, er den del af systemet der styrer kommunikationen mellem vægten og databasen. ASE'en er således et mellemled imellem en operatør, der står ved vægten og afvejer råvarer, og databasen.

Vi har tidligere i CDIO2 i kurset Datakommunikation udviklet en ASE men den viste sig at være meget mangelfuld i forhold til dette projekts funktionelle krav så derfor valgte vi at starte helt forfra med den i dette projekt.
Følgende afsnit vil kort beskrive hvordan ASE'en er designet ud fra et klassediagram samt analysere hvorvidt dette design stemmer overens med 3-lags modellen. Yderligere vil afsnittet kort skitsere forskellige test af ASE'en for til sidst at skitsere afgrænsninger og forbedringer der kunne laves.

Analyse af ASE

Ud fra det udleverede projektoplæg kunne vi udlede hvilke krav der var til vores ASE. Det største krav var, at den skulle overholde en bestemt afvejningsprocedure der var angivet i oplægget og som kan læses forneden:

Afvejningsproceduren

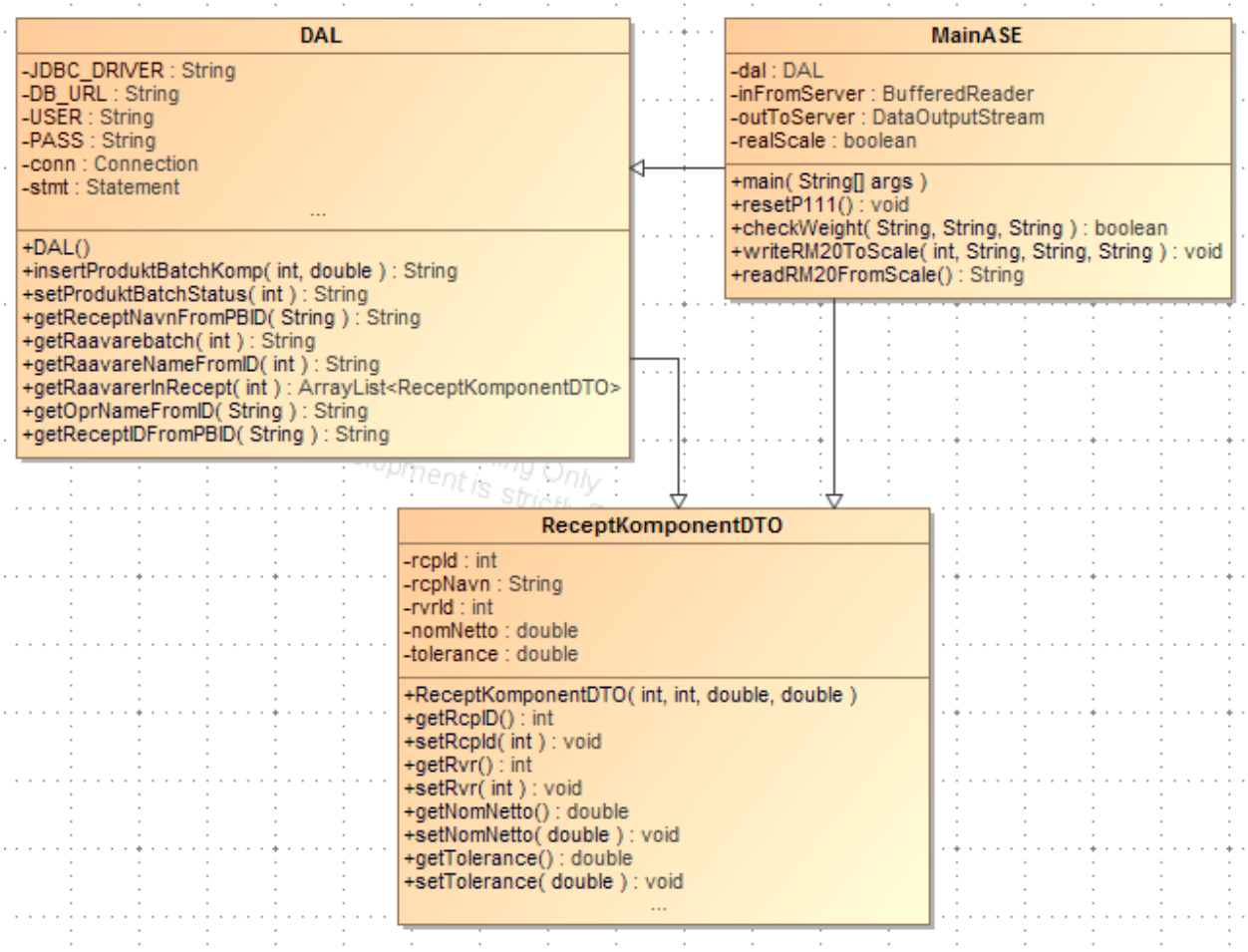
1. Operatøren har modtaget en produktionsforskrift på papir fra værkføreren.
2. Operatøren vælger en afvejningsterminal.
3. Operatøren indtaster operatør nr.
4. Vægten svarer tilbage med operatørnavn som så godkendes.
5. Operatøren indtaster produktbatch nummer.
6. Vægten svarer tilbage med navn på recept der skal produceres (eks: saltvand med citron)
7. Operatøren kontrollerer at vægten er ubelastet og trykker 'ok'
8. Systemet sætter produktbatch nummerets status til "Under produktion".
9: Vægten tareres.
10. Vægten beder om første tara beholder.
11. Operatør placerer første tarabeholder og trykker 'ok'
12. Vægten af tarabeholder registreres
13. Vægten tareres.
14. Vægten beder om raavarebatch nummer på første råvare.
15. Operatøren afvejer op til den ønskede mængde og trykker 'ok'
16. Pkt. 7 – 14 gentages indtil alle råvarer er afvejet.
17. Systemet sætter produktbatch nummerets status til "Afsluttet".
18. Det kan herefter genoptages af en ny operatør.

Umiddelbart ser det nemt nok ud at lave og det er det sådan set også hvis der aldrig kunne gå noget galt. Problemet er blot, at der altid kan gå noget galt når der sidder et menneske i den anden ende. Man kunne f.eks. forestille sig at en operatør i trin 3 i afvejningsproceduren indtastede et operatørID der ikke fandtes i databasen. I sådan et tilfælde skal systemet ikke crashe men komme og spørge om et nyt ID.

Denne slags input-validering er der rigtig meget af i løbet af afvejningsproceduren og således er det lige pludselig en knapt så simpel opgave at udvikle.

Design af ASE

Nedenfor ses figur 2 der er klassediagrammet for ASE'en.



Figur 2: Klassediagram for ASE

Det ses på figur 2 at der ikke er mange klasser involveret i denne applikation. Det er fordi, at ASE i bund og grund er en ret simpel applikation. Den har brug for et datalag, DAL, til at hente og skrive data til databasen. Til at gemme data komponenter fra databasen bruges en ArrayList af ReceptKomponentDTO'er. En instans af ReceptKomponentDTO svarer til en række(tupel) i databasen. Vi brugte denne tidlige analytiske model til at gå videre til designfasen og implementation af ASE.

Implementation af ASE

Umiddelbart, som det fremstod i analysen af ASE, kunne det være en simpel opgave at udvikle ASE men det viste sig hurtigt at være problematisk. Forneden, på figur 3, vil vi kort gennemgå trin 3 i afvejningsproceduren for at vise et udsnit af den kode, der er bag.

```
// Prompt for gyldigt operatør nummer på vægt
do {
    writeRM20ToScale(4, "Operator ID?", "", "");
    opr_id = readRM20FromScale();
    opr_name = dal.getOprNameFromID(opr_id);
    // Skriv P111 hvis ID ikke findes (og vent 2 sekunder så bruger kan nå at læse)
    if ("ID findes ikke!".equals(opr_name) || "SQL fejl".equals(opr_name)) {
        outToServer.writeBytes("P111 \"ID findes ikke - Proev igen!\"" + '\n');
        System.out.println(inputServer.readLine());
        if (!realScale) inputServer.readLine();
        Thread.sleep(2000);
    }
    else {
        resetP111();
    }
} while ("ID findes ikke!".equals(opr_name) || "SQL fejl".equals(opr_name));
```

Figur 3: Uddrag af ASE implementation

Vi starter med at lave et loop fordi vi som sagt ikke ved hvor mange gange der skal spørges om et operatør ID - dette afhænger af hvad brugeren indtaster. Dernæst skriver vi en RM20 kommando til vægten - RM20 kommandoen er en kommando vægten forstår og den står for at spørge om input fra brugeren og derefter sende det tilbage til vores applikation.

Når operatøren indtaster et ID på vægten, aflæser vi dets værdi og kalder metoden “getOprNameFromID(opr_id)” på datalaget. Dette er en metode i datalaget der kører en SQL forespørgsel i databasen og fortæller hvorvidt der findes en operatør med det ID og i så fald returneres dennes navn. Hvis ID ikke findes returneres “ID findes ikke!”, hvilket resulterer i at der skrives en statusbesked på vægten (en P111 besked) der fortæller at ID ikke findes. Loopet vil herefter køre en iteration mere siden at variablen “opr_name” er “ID findes ikke!”.

Dette er kun for trin 3 af afvejningsproceduren og det kan derfor ses at det meget hurtigt bliver omstændigt at lave en fuld afvejningsprocedure med input-validering i ASE. Vi har altså brugt en del tid i projektperioden på at få denne komponent af systemet op at køre.

ASE og 3-lags model

Den ASE vi har udviklet i dette projekt, er opbygget med et Data Access Layer, DAL, men derudover ligger alt i én Main klasse. Denne Main klasse fungerer som et funktionslag, da al funktionalitet ligger i denne klasse. ASEn har ikke noget View-funktionalitet, fordi en bruger ikke har direkte interaktion med den, men blot bruger den eksternt. Derfor kan der argumenteres for at ASEn overholder 3-lags modellen på trods af, at den kun har et data lag og et controller lag. Man kan sige at funktionslaget og viewlaget er smeltet sammen til ét.

Den ASE vi har udviklet overholder ikke 3-lags modellen. Vi mener ikke at det giver mening at komplicere et program unødigt, og vores program er endt som en 2-lags model, hvor man kan sige at boundary og funktonslaget er smeltet sammen. Vores datalag står stadig uafhængigt grundet exception-handling.

Test ASE

Der er udført en række tests på afvejnings styringsenheden, der er dokumenteret herunder. Alle tests af ASE er forgået via vægt-terminalen, da denne virker som brugerflade for programmet. Alle tests af ASE er tests af implementering af use case 6.

Test case: Indlæsning af operatør
Der testes for følgende funktioner ved indlæsning af operatør: ● At det korrekte operatør navn vælges ud fra det bruger-agivne operatør ID ● At der ved angivning af en en ikke-eksisterende operatør efterfølgende anmodes om nyt operatør ID **Forudsætninger:** ● Databasen er tilgængelig under hele testen ● Forbindelsen mellem ASE og vægt-terminalen er gyldig under hele testen. **Procedure:** Ved vægt-terminalen angives det at der skal indtastes et ID før end at afvejnings processen skal startes. Ved brug af terminalens input indtastes et ID. **Resultater:** Ved korrekt ID angiver vægt-terminalen efterfølgende, at operatøren skal validere sit navn for at fortsætte afvejningsprocessen, hvilket er korrekt. Ved et ID der ikke eksisterer i databasen angiver vægt-terminalen, at der skal indtastes et nyt operatør ID, hvilket er korrekt opførsel. Ved efterfølgende indtastning af både ugyldigt og gyldigt ID fortsætter processen ved korrekt at bede om hhv. et korrekt ID eller bede om navn.

Idet systemet beder operatøren om at validere sit navn, er der kun to muligheder, at sige ja eller nej.

Ved et ja går systemet korrekt videre til at anmode operatøren om at indtaste en produktbatch ID. Ved et nej beder systemet korrekt om at man indtaster et nyt operatør ID. Processen starter så forfra, hvor man skal indtaste et gyldigt ID og efterfølgende validere sit navn.

Konklusion: Testen er gennemført med succes uden fejl.

Test case: Indlæsning af produktbatch

Der testes for følgende funktioner ved indlæsning af en produktbatch:

- At den korrekte produktbatch vælges ud fra det bruger-angivne produktbatch ID, den tilhørende raavare batch korrekt vises på displayet, og at selve afvejningsprocessen starter efterfølgende.
- At operatøren ved indtastning af et ugyldig ID promptes på ny for at indtaste et gyldigt ID.

Forudsætninger:

- Databasen er tilgængelig under hele testen
- Forbindelsen mellem ASE og vægt-terminalen er gyldig under hele testen.
- Operatør er indlæst korrekt og navnet valideret.

Procedure:
Efter indlæsning af en gyldig operatør promptes operatøren på vægt-terminalen til at indlæse en produktbatch. Dette gøres ved at operatøren indtaster et produktbatch ID. ASE kontrollerer, om dette ID eksisterer i databasen. Gør det ikke det, skal operatøren promptes til at indtaste et gyldigt. Hvis ID'et derimod eksisterer, får operatøren at vide hvad batchen hedder, og selve afvejningsprocessen starter.

Resultater:
Ved indtastning af et gyldigt ID skriver vægt-terminalen korrekt navnet på råvare batchen på terminalen, og afvejnings processen starter umiddelbart efter.

Ved indtastning af et ugyldigt ID bliver operatøren via terminalen gjort opmærksom på at det indtastede ID ikke eksisterer i databasen, og bliver på ny anmodet om at indtaste et gyldigt ID.

Konklusion: Testen er gennemført med succes uden fejl.

Test case: Afvejningsprocessen

Der testes for følgende funktioner under afvejningsprocessen:

- Før hver afvejning promptes operatøren til at kontrollere, at vægten er ubelastet.
- At systemet, idet afvejning af produktbatchen starter, sætter den pågældende produktbatch i databasen som værende under produktion.
- Vægten tareres før og efter beholder placeres på vægten før hver afvejning.
- Vægten prompter operatøren om at indtaste råvarebatch ID på hver råvare indtil alle råvarer er afvejet.
- Vægten fortsætter processen idet en afvejning er korrekt, dvs. efter operatøren har afvejet den korrekte mængde af den angivne råvare, forsætter til den næste råvare.
- At systemet gennemgår alle de korrekte råvarer.
- Efter afvejning af alle råvarer korrekt angiver den pågældende produktbatch som værende produceret i databasen.
- Systemet korrekt går tilbage til at bede operatøren om at identificere sig igen for at starte en ny batch.
-

Forudsætninger:

- Databasen er tilgængelig under hele testen
- Forbindelsen mellem ASE og vægt-terminalen er gyldig under hele testen.
- Operatør er indlæst korrekt og navnet valideret.
- Produktbatch er indlæst korrekt

Procedure:
Operatøren kontrollerer at vægten er ubelastet, og sender “OK” til ASE via terminalen. ASE prompter herefter operatøren til at indtaste den første råvarebatch ID. Ved gyldig råvarebatch prompter terminalen operatøren til at stille en beholder på vægten, og derefter tarere vægten. Herefter skal operatøren afveje en korrekt mængde af den angivne råvare. Der testes for for lidt af den angivne råvare, for meget og tilpas mængde. Herefter angiver vægten at afvejningen af den pågældende råvare er tilpas, og fortsætter efterfølgende med den næste råvare.
Efter alle råvarer er afvejet korrekt bliver operatøren via terminalen gjort opmærksom på, at afvejningen er gennemført.

Resultater:
Både ved ubelastet og belastet vægt forsætter afvejningsprocessen. Dette giver mening, da operatøren kunne have en god grund til at lade vægten være belastet under hele afvejningen.

Det blev observeret via. overvågning af databasen i MySQL-Workbench under testen, at databasens informationer om hvorvidt batchen var i produktion korrekt blev opdateret under afvejningen.

Ved indtastning af gyldigt råvarebatch ID fortsætter vægten med afvejning af den pågædende råvare. Ved ugyldigt ID bliver operatøren gjort opmærksom på at råvaren ikke er fundet i databasen.

Ved afvejning af for lidt eller for meget af den pågældende råvare forsætter vægten ikke afvejningen. Dvs. at vægten korrekt afventer at den afvejede mængde er korrekt.

Ved slut af afvejningen gik systemet korrekt tilbage til at anmode om at operatøren logger ind,

der betyder at processen er afsluttet. Det blev ligeledes observeret efter testen at systemet i databasen korrekt opdaterede produktionsstatus for batchen som værende færdig.

Konklusion: Testen er gennemført med succes uden fejl.

Afgrænsning for ASE

I forbindelse med udviklingen af ASE har vi måtte tage nogle valg i forhold til hvilke funktioner vi ville implementere. Projektperioden varer ikke for evigt og således har vi simpelthen måtte indse, at det ikke er alle funktionaliter og småfejl vi kan nå at rette. Det været nødvendigt at prioritere visse funktioner samt at skære ned på andre. I denne afgrænsning vil vi kort komme ind på eventuelle funktioner vi ikke har implementeret samt eventuelle fejl vi har fundet i programmet men som der ikke har været tid til at rette.

- Når der læses den nuværende vægt (masse) fra den fysiske vægtterminal, får ASE det tilbage med 3 decimaltal (f.eks. 2.346 kg). Denne aflæste vægt skal sammenlignes med den nominelle netto vægt der ønskes for en given råvare plus/minus den angivne tolerance i procent. Denne vægt med udregnet tolerance har flere end 3 decimaler (f.eks. 2.34667 kg) og således kan det give problemer når vi forsøger at sammenligne de to tal. Dette er en irriterende fejl men vi har ikke fundet tiden til at rette den, da man kan klare afvejninger uden at rette op på den.

- Som en forbedring til ASE bør man også tjekke, om en given produktbatchkomponent allerede er produceret før en operatør begynder at producere den. I denne endelige iteration af programmet bliver det ikke tjekket og således kan en operatør réelt producere en produktbatchkomponent der allerede er produceret.

- Hvis den fysiske vægt overbelastes (der placeres en masse over 6 kg), crasher ASE når den forsøger at aflæse vægten. Dette er et problem vi ikke har kunnet løse på den korte tid vi havde tilbage til fejlretning.

- Hvis den fysiske vægt tareres og man fjerner den beholder der står på således at vægtens reélle masse er f.eks. -0.336 kg, så crasher ASE når den forsøger at aflæse vægten.

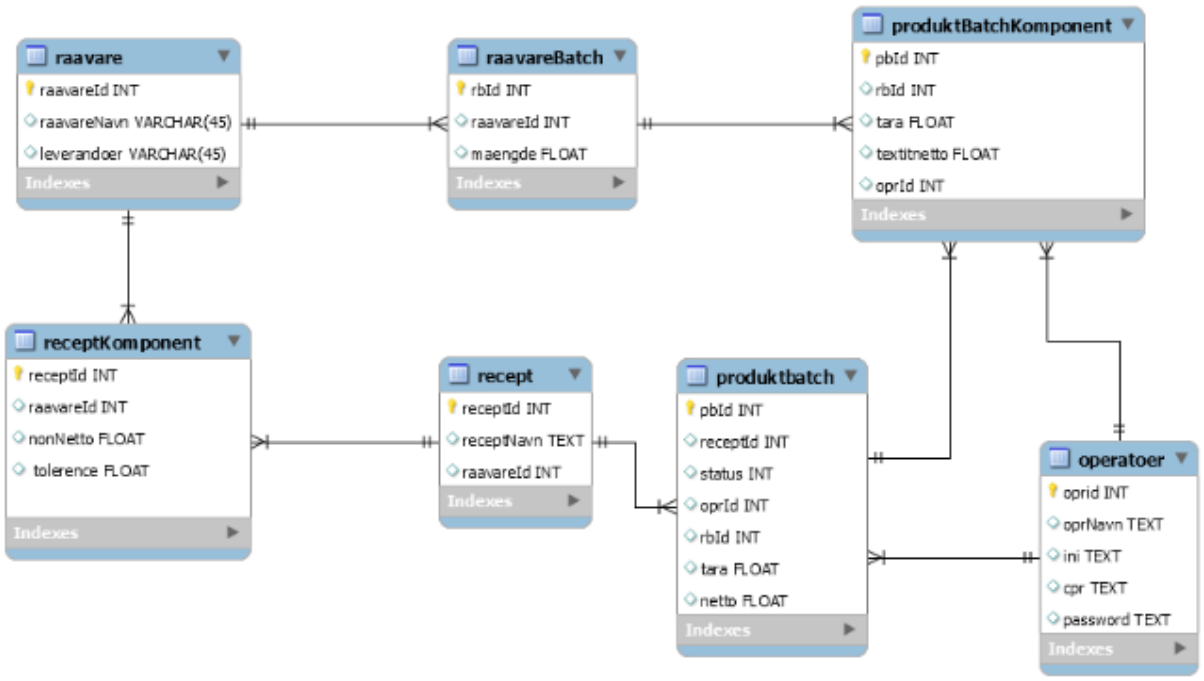
Delkonklusion ASE

Alt i alt har vi fået udviklet en funktionsdygtig Afvejnings Styrings Enhed som kan bruges til at producere produkter. Man kan komme som operatør med en produktionsforskrift i hånden og indtaste sit ID og derefter en produktbatch ID og derefter klarer ASE resten for en.

Den opfylder således kravene for afvejningsproceduren. Det har ikke været den nemmeste opgave i og med at der har været meget input-validering men i sidste ende har vi udviklet et udmærket produkt.

Databasen

Som vist i introduktion af systemet fungerer databasen som en slags bindeled mellem GWT og ASE'en og vægten. Istedet for at repetere konklusioner fra rapporten i (02327) så er formålet med følgende afsnit at forklare, meget kort, hvilken database model vi har valgt at gå med samt hvordan kommunikation mellem GWT og ASE fungerer. Forneden på figur 4 ses et ER-diagram over vores database model:



Figur 4: ER diagram for database

Det kan ses foroven hvordan vores database er opbygget. Diagrammet viser, hvilket tabeller vi har i databasen samt hvilke attributter de har hver især. I forhold til projektoplægget bemærkes det, at vi har valgt at tilføje "receptkomponent" og "produktbacthkomponent" entiteter for at overholde tredje normalform og undgå redundans i databasen.

Databasen og GWT

Datalaget for GWT i vores projekt er en enkelt klasse kaldet 'DAO', hvilket står for Data Access Object. Det er i denne klasse, at applikationen forbinder sig til selve databasen. Når forbindelsen er oprettet, kan man ændre i databasen via SQL kode. Derfor skal kommandoerne man skriver i denne klasse foregå i SQL kode, da det er dette sprog som databasen forstår. SQL koderne bliver gemt som metoder, som let kan bruges i andre klasser ved at importere DAO'en og lave en instans af den.

Når selve metoderne og forbindelsen er sat på plads, kan man lave nye objekter eller rette på dem via DTO'er. En DTO repræsenterer et enkelt objekt i én af entiteterne i databasen, altså en række eller tupel.

Databasen og ASE

For ASE gør det sig ligeledes gældende, at der findes en klasse kaldet DAL - Data Acces Layer. Denne fungerer ligesom DAO beskrevet foroven. ASE opretter ved initiering en instans af dette DAL objekt, som siden hen kan bruges til at udhente og skrive informationer til/fra databasen.

CDIO Final - GWT

Indledning

En vigtig del af systemet var at lave en brugervenlig brugerinterface, der skulle kunne tilgås af alt fra superbrugere, værkførere til farmaceuter. Til det, var det et krav, at vi brugte Google Web Toolkit (GWT).
Nedenfor vil vi gennemgå hvordan GWT fungerer samt hvordan vores webapplikation blev udviklet fra analyse stadie med kravsspecifikation og usecases til design fasen med klassediagram og implementation og test.

GWT

GWT er en relativt ny udviklingsmetode inden for web programmering, der især udmærker sig ved at gøre browser orienterede applikationer lettere at programmere. I projektet har vi hovedsageligt arbejdet med Java, som GWT oversætter til JavaScript. GWT fungerer dermed i høj grad "behind the scene".

Mulighederne ved GWT

GWT står for Google Web Toolkit og er et open-source toolkit der kan bruges til at udvikle browser-baserede applikationer. Det tillader udvikleren at lave web applikationer uden at være en ekspert i alle de mange forskellige browsere, HTML, CSS eller JavaScript[1]. Alt dette kan GWT hjælpe en med at lave ved at auto-generere mange af de ting der ligger inde bag ved ens hjemmeside.
Undervejs i kurset har vi stiftet bekendtskab med denne GWT teknologi og vi har afleveret et CDIO projekt hvor vi fik rigtig godt kendskab til det grundlæggende teori bag en GWT webapplikation.

Det eneste man skal kunne for at udvikle en hjemmeside med GWT er Java. GWT lader en skrive programmer i Java som siden hen bliver implementeret som højt optimeret JavaScript, der kan køre på alle browsere. At kunne en lille smule CSS og HTML er også hensigtsmæssigt, så man kan sætte sin hjemmesides design op men det er ikke en nødvendighed for at lave en fungerende hjemmeside.

GWT indeholder en masse standard widgets og komponenter som meget nemt kan bruges til at designe og implementere sin hjemmeside. Nævneværdige widgets der er gjort brug af i dette projekt er Button, TextBox og FlexTable. GWT applikationer kører som JavaScript i en brugers internet browser. Der kan dog af og til være brug for at applikationen kan kommunikere med f.eks. en database-server, som i dette projekt. Dette gøres ved hjælp af Remote Procedure Calls, også kendt som RPC.

Remote Procedure Call

RPC står for udvekslingen af Java objekter mellem klient og server delen af en applikation. Den server-side kode der bliver kørt når klient-siden forespørger noget data, kaldes en service.
Når der foretages et RPC er det et asynkront kald. Det vil sige at programmet sender kaldet og så kører det videre, uden at vente på svaret. Det er altså ikke et “blocking” call. Når svaret så er klar fra serveren, sendes det tilbage til klienten der så behandler det. Der er flere fordele fra et slut-bruger perspektiv ved denne form for kald. Den største fordel er, at brugergrænsefladen forbliver responsiv, den “hænger” ikke mens der ventes på et svar fra en server.

For at lave et RPC kald, skal man implementerer et par interfaces der er defineret i GWT. Man skal først definere et interface til ens service. Dette interface skal extend RemoteService, der er en del af GWT, og i dette interface skal alle de metoder som skal kunne kaldes som RPC være skrevet op.

[1] http://www.gwtproject.org/overview.html

Dernæst skal der være en klasse der extends RemoteServiceServlet og implementerer det interface der er beskrevet ovenover. Endeligt skal der laves et asynkront interface til ens service, der kaldes fra klient-sidens kode. På denne måde, ved hjælp af GWT's RemoteServiceServlet, håndteres automatisk data der passerer mellem klient og server delen af hjemmesiden og det sikrer, at den rigtige metode på serveren kaldes når der kaldes en metode på klienten.

Analyse

I analyse afsnittet vil vi redegøre for det indledende fase af projektet. Afsnittet vil således starte med at skitsere de kravspecifikationer vi har tolket ud fra projektoplægget og derefter vil vi komme ind på use-cases i forhold til disse krav.

Kravspecifikationer

En vigtig del af projektarbejde er at udarbejde kravspecifikation tidligt i projektfasen. I forbindelse med dette projekt fik vi udleveret et oplæg der har fungeret som kontrakt med kunden. I udarbejdelsen af projektet har vi foretaget mindre ændringer som f.eks ændring af databasen. Nedenfor ses de funktionelle krav for projektet, vi vil i Design afsnittet kort komme ind på hvilke af disse krav vi dog ikke helt har opfyldt ved denne iteration af CDIO final.

Funktionelle krav:

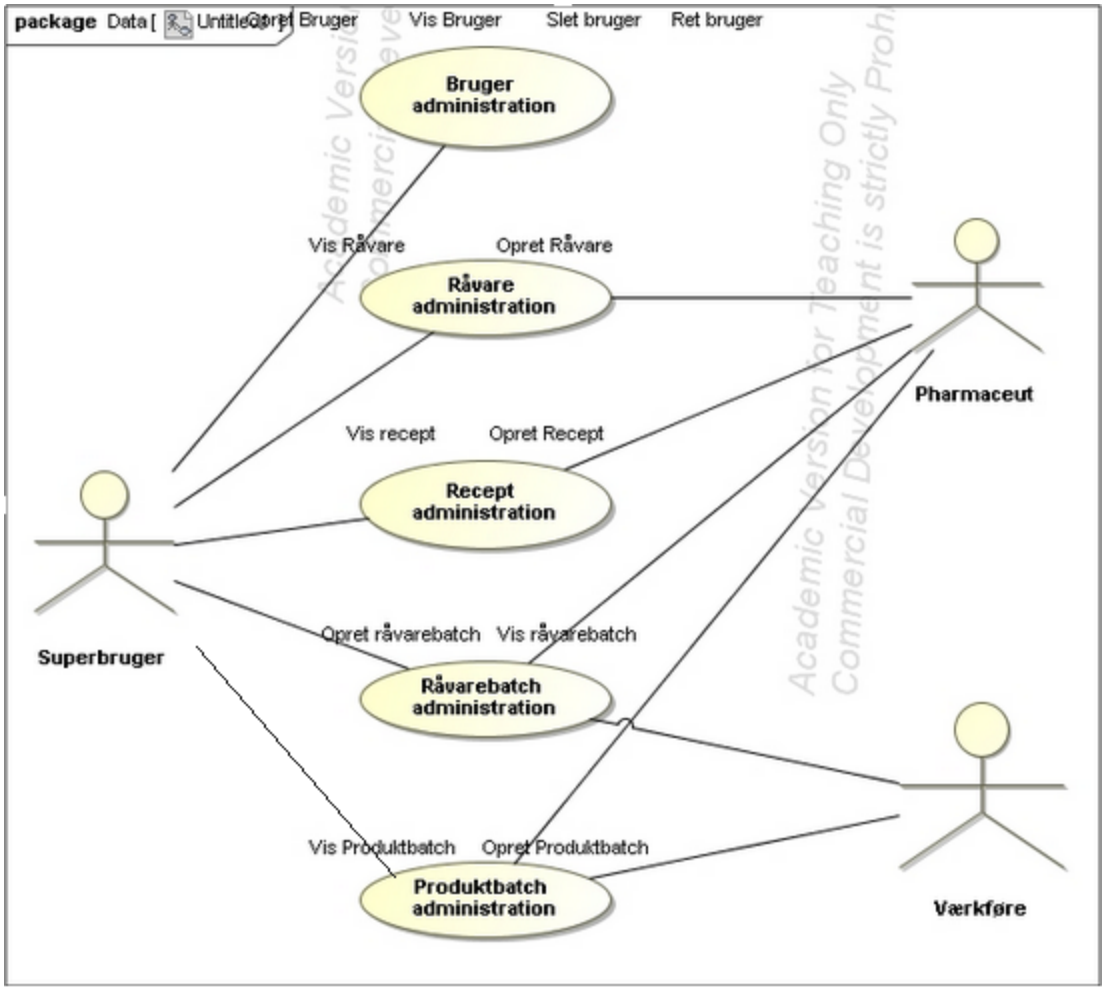
1. Der skal oprettes en database. Denne skal følge kravene fra oplægget
2. Datalaget skal implementeres som en SQL database.
3. Der skal udvikles en GWT applikation med følgende funktioner:
    - Oprette, slette, vise og rette brugere
    - Oprette og vise råvarer
    - Oprette og vise råvarebatches
    - Oprette og vise recepter
    - Oprette og vise receptkomponenter
    - Oprette og vise produktbatches
4. Man skal kunne logge ind som en bestemt type bruger og ud fra brugertypen kunne forskellige dele af ovenstående funktioner

Ikke-funktionelle krav:

1. Databasen skal minimum være på 3. normalform
2. Programmet skal kunne køres på computerne på DTU's databarer
3. Programmet skal skrives i Java

Use Case Diagram

Use Case diagrammer er et vigtig element i at få et hurtigt overblik over hvilken funktion der knytter sig til systemet. Vi havde i projektoplægget fået udleveret 6 forskellige use-cases. Ud af de seks knyttede 5 sig til brugen af GWT webapplikationen. Nedenfor, på figur 5, har vi illustreret use-casene som et UML diagram. I bilag kan man finde en mere detaljeret gennemgang af hver enkelt usecase, som en del af dokumentationen for systemet.



Figur 5: Use-case diagram for GWT

Ovenfor ses Use case diagram for GWT systemet. Man kan inddele brugerne i 3 forskellige niveauer med forskellige beføjelser. Nedenfor vi vi beskrive de forskellige usecases i nærmere detaljer.

Værkføreren

Den laveste i hierakiet er en værkfører. Han har kun muligheden for 2 forskellige use cases, produkt- og råvarebatch administration. Værkførens arbejde består i at bestille varer hjem og fylde dem på laget i råvarebatches samt at oprette produktbatches som operatører kan producere.

Farmaceuten

I midten af hierakiet befinder sig farmaceuten. Han har rettigheder til at udføre det samme som en værkføreren, men kan udover det også administrere råvarer og recepter. Farmaceutens arbejde er derfor, at oprette de forskellige råvare der skal bruges, i systemet, og derefter kan disse bruges til at oprette nye recepter.

Superbrugeren

Øverst i hierakiet er Superbrugeren. Han har samme rettigheder som farmaceuten men kan derudover også administrere de forskellige brugere i systemet. Det er altså hans arbejde at styre de forskellige personers arbejdsposition.

Det er værd at notere at superbrugeren har en use case 'slet bruger', men denne sletter faktisk ikke brugeren fra selve databasen over de forskellige brugere. I stedet bliver brugerens attribut 'aktiv' sat fra 1 til 0, hvor 0 repræsenterer at brugeren ikke længere er en aktiv del af firmaet. Dette er gjort for at bibeholde sporbarhed i databasen.

Domænemodel

Vores domænemodel har vi valgt at dele op i 2, da vi bruger mange klasser for at opretholde en fornuftig struktur i vores GWT. Vi beskriver først hvad hver enkelt package har, og derefter beskriver vi hvordan de forskellige packages "snakker" med hinanden - altså deres forhold hvis man kan sige det sådan.

Ved at gøre det på denne måde, mister vi evnen til at vise præcis hvilke klasser der kommunikerer med hvilke, men vi følte det var nødvendigt, da det ellers ville være alt for uoverskueligt at beskrive alle vores potentielle klasser i GWT applikation.

UI-package

Vi starter med at gennemgå de forskellige packages. Den første med flest klasser er UI-packagen, som ses på bilag 7. Her er ContentView øverst, og det er denne der har alle menuerne og alle metoderne. Denne leder til forsiden, hvor man logger sig ind. Alt efter hvilket

niveau man har, kommer man ind på én af tre menuer. Inde på denne menu man bliver logget ind på, kan man se hvad slags arbejde man har lov til at udfører via de forskellige knapper. Knapperne repræsenterer use cases som leder hen til specifikke metoder.

På bilaget har vi forkortet det, og kun vist hvad de forskellige bruger må udfører, som andre under deres niveau ikke må. Altså har pharmaceuten på modellen 3 use cases, når den i virkeligheden har 5, hvilket er dem som værkføren også har. Det skal desuden nævnes at vi har splittet recept op i 2, recept og receptkomponent.
Resten af bilagene 8 til 13 er ikke specielt spændende, da de ikke har samme komplekse struktur. Vi vil derfor forklare, hvordan disse packages kommunikere med hinanden. Selve domænediagrammet er på bilag 14, hvor vi vil starte på toppen med pakken 'client'.

Client-package

Client indeholder en enkelt klasse kaldet 'kartotek'. Denne klasse implementere EntryPoint i GWT, hvilket er den klasse, GWT bruger som "start-klasse". Man kan altså sige det er GWT's main klasse der bliver opsat her. Den kalder MainView's run metode, hvilket skaber en ensidet forbindelse mellem Client og Controller, da MainView ligger i Controller.

Fra Controllerens klasse MainView kalder man så metoden openWelcomeView så man kommer ind i klassen WelcomeView, hvilket skaber en forbindelse mellem Controller og UI'en. Denne forbindelse går begge veje, da WelcomeView henter metoderne der er i MainView for at komme videre. Altså skal der være en dobbeltforbindelse mellem de 2 packages Controller og UI.

Controlleren laver også en instans af DatabaseServiceClientImpl klassen, som opretter siden. Der er ingen kald tilbage til samme klasse, så dette er en enkelt sidet forbindelse.Der er ikke flere forbindelser fra eller til Controller klassen.

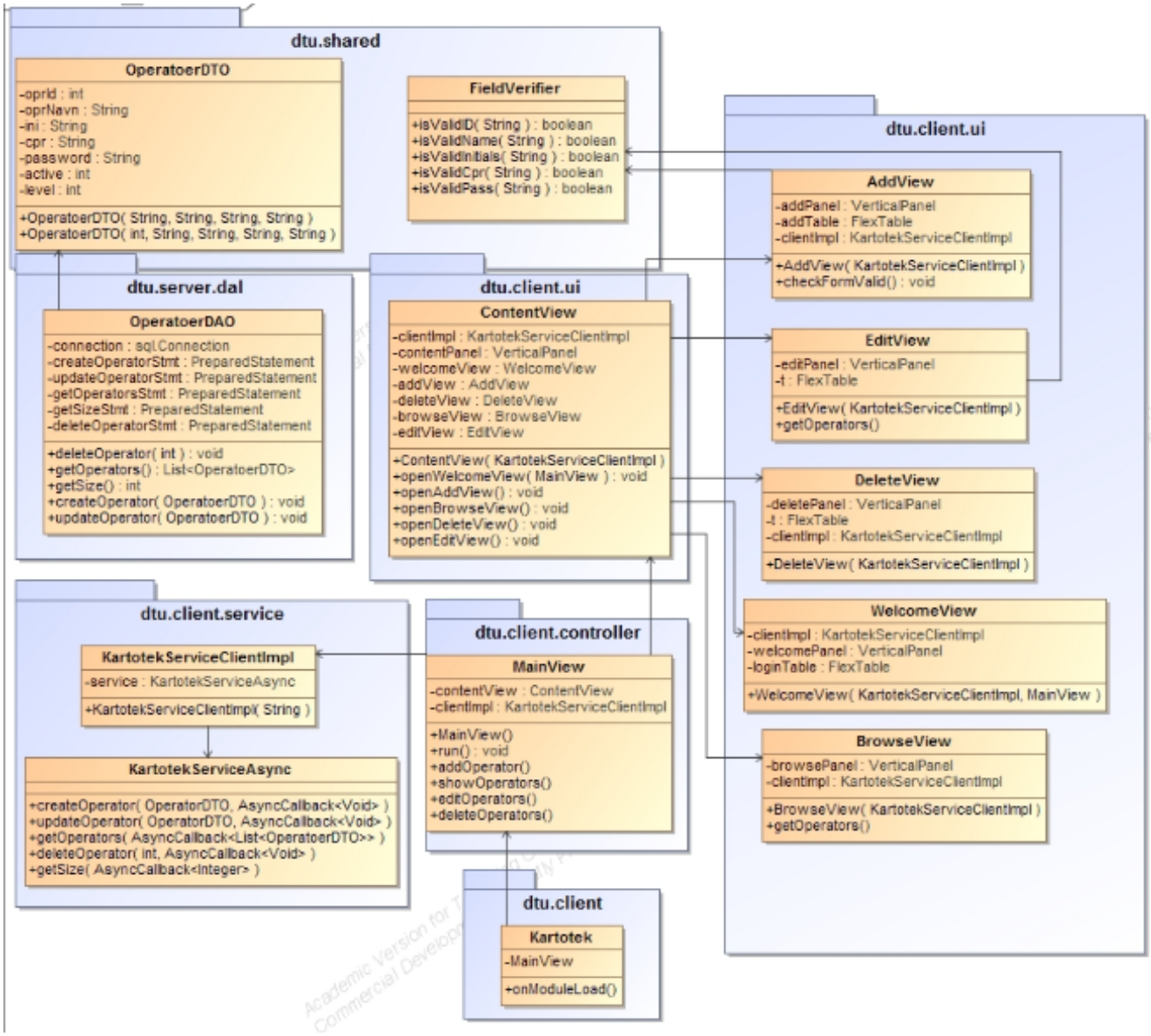
Den næste vi kigger på er UI pakken. Denne pakke indeholder alt det, som bruger skal bruge til at udfører de forskellige use cases. Denne kan ses detaljeret i bilag 7. Denne er central for programmet og forbinder sig til alle pakkerne end clienten. Den forbinder sig til:

- Service via WelcomeView. Denne peger kun mod Service
- Shared igennem alle de forskellige metoder, såsom AddUser. Dette sker nede i DTO'erne, som hver har deres egen struktur alt efter hvilken entitet de hører til
- DAL'en via alle de funktioner som opretter nye objekter i entiterne i databasen.
- Controller igennem de forskellige menuer der ligger i pakken.

Den sidste pakke der refere til andre pakker, er Service. Denne fungere som et interface til de forskellige metoder som skal udfører det som brugeren foretager sig.

Design - Klassediagram

Vi har i forbindelse med projektet udviklet et design klassediagram. Klasserne i diagrammet er inddelt i pakker, ligesom de er i Java projektet. Dette er gjort for at skabe et bedre overblik over hvordan disse klasser afhænger af hinanden samt hvor de hører til.
Det skal dog nævnes at vi har valgt ikke at lave et klassediagram over samtlige klasser i projektet da dette ville blive alt for omstændigt og uoverskueligt at håndtere på én gang. Vi har derfor valgt at tage udgangspunkt i en enkelt use-case, nemlig den hvor der kan oprettes brugere. Forneden ses design klassediagram for dette:



Figur 6 - Klassediagram for GWT

For at knytte et par ord til ovenstående figur, så viser det hvilke klasser der findes i projektet i forhold til oprettelse af brugere, samt hvordan de gør brug af hinanden. I GWT skal der angives en Entry point klasse der i dette eksempel er Kartotek klassen. Når denne bliver initialiseret, opretter den en instans af MainView, der er controller for hele GUI'en.

MainView instantierer et KartotekService objekt som sendes med videre rundt til alle views der oprettes, således at de kan gøre brug af kommandoer til og fra server delen af applikationen. Det er via denne KartotekService at klient og server del af applikationen kan snakke sammen via de tidligere omtalte RPC asynkrone kald.

Klassediagrammet foroven var et tidligt design klasse diagram og således har vores reélle implementation af projektet ændret sig en smule i forhold til dette tidlige diagram. Undervejs i projektforløbet har vi gjort brug af udviklingsmetoden "Unified Proces" som bygger på en inkremental og iterativ proces og således bliver man hele tiden "klogere" efterhånden som projektet skrider frem og derfor afspejler diagrammet foroven ikke 100% det, vi er endt op med.

Når først GWT applikationen startes, starter applikationen i EntryPoint klassen der er defineret i GWT's indstillinger - XML filer. I vores kode er EntryPoint Kartotek klassen. Derfra oprettes instanser af de andre objekter. Hiearkiet af dette kan ses i klassediagrammet på figur 6.

Implementation

Forneden, på figur 7, vil vi kort gennemgå en del af klassen UserBrowseView for at vise et udsnit af den kode, der er bag en side i vores GWT applikation. UserBrowseView er den klasse der er ansvarlig for at vise de brugere der forefindes i databasen. - se næste side

```
private void getUsers() {
    clientImpl.service.getUser(new AsyncCallback<List<UserDTO>>() {

        @Override
        public void onFailure(Throwable caught) {
            Window.alert("Server fejl!" + caught.getMessage());
        }

        @Override
        public void onSuccess(List<UserDTO> result) {
            int j = 1;
            for (int i=0; i < result.size(); i++) {
                if (!showInactive) {
                    if (Integer.valueOf(result.get(i).getActive()) == 1) {
                        t.setText(j, 0, ""+result.get(i).getOprId());
                        t.setText(j, 1, result.get(i).getOprNavn());
                        t.setText(j, 2, result.get(i).getIni());
                        t.setText(j, 3, result.get(i).getCpr());
                        t.setText(j, 4, result.get(i).getPassword());
                        t.setText(j, 5, result.get(i).getActive());
                        t.setText(j, 6, result.get(i).getLevel());
                        j++;
                    }
                }
                else {
                    t.setText(j, 0, ""+result.get(i).getOprId());
                    t.setText(j, 1, result.get(i).getOprNavn());
                    t.setText(j, 2, result.get(i).getIni());
                    t.setText(j, 3, result.get(i).getCpr());
                    t.setText(j, 4, result.get(i).getPassword());
                    t.setText(j, 5, result.get(i).getActive());
                    t.setText(j, 6, result.get(i).getLevel());
                    j++;
                }
            }
        }
    });
}
```

Figur 7 - Uddrag af GWT kode

Ovenfor ses figur 7 der er et udsnit af klassen UserBrowseView. Figuren viser en metode som bruges til at indhente data fra databasen og indsætte det i en FlexTable som så kan præsenteres for brugeren. Dette gøres ved at kalde getUser() metoden, der er en såkaldt service der er defineret i interface mellem klient-delen af GWT og server-delen. Når der kaldes en getUser() metode fra brugergrænsefladen (klient-delen), sender GWT kommandoen videre til sever-delen som så kan hente data ud fra databasen, vha. den føromtalte DAO klasse. Dette foregår ved hjælp af asynkrone Remote Procedure Calls. Da det er asynkrone kald, vides ikke hvornår svaret kommer tilbage. Derfor oprettes en handler internet i GWT der venter på svaret fra serveren. Når

svaret endelig kommer, kan udfaldet være to ting. Kaldet kan være gået succesfuldt ovre på server-siden (hvilket betyder at der ikke er gået noget galt og er kastet en exception) og i dette tilfælde vil koden defineret i onSuccess() metoden blive kørt. Hvis service kaldet af en eller anden årsag er fejlet, køres onFailure() koden.

Hvis onSuccess() koden køres, ses det at vi tilføjer en masse ting til et objekt kaldet "t". Dette objekt "t" er af typen FlexTable, der er en standard widget i GWT. Man kan betragte det som en dynamisk tabel hvor man frit kan indsætte og fjerne elementer i rækker og kolonner. De er rigtig smarte til at lave layout og rigtig nemme at arbejde med.

Ovenstående var en meget kort gennemgang af hvordan man på klient-siden henter data fra en server ved hjælp af RPC. Det er samme princip vi gør brug af i rigtig mange af klasserne og således har vi kun vist et kort eksempel fra en af klasserne, for ikke at gentage os selv alt for meget.

GWT og 3-lags modellen

Vi har tidligere stiftet bekendskab med 3-lags modellen i forbindelse med vores projektarbejde i løbet af kurset. 3-lags modellen er enmåde hvorpå man strukturerer koden således at projektet let kan rettes og der kan udbygges i systemet. Dette sker formelt set ved at opdele koden i et boundary, controller og datalag. Denne form for opdeling er desværre ikke i samme grad mulig at følge ved brug af GWT. Nedstående afsnit vil argumentere for hvordan vores kode er opdelt, samt hvorfor 3-lagsmodellen ikke direkte kan overføres til GWT.

Som tidligere nævnt forsøger 3-lags modellen at inddele et program i 3 lag, nemlig et datalag, et user-interface lag, og et controller lag.
Vores program opfylder to af disse kriterier. Datalag er knyttet til databasen, hvor user interfacet eksisterer i form af GWT's forskellige metoder til at lave interfaces. Der hvor programmet adskiller sig fra 3-lagsmodellen er i vores controllers, som ligger i samme lag som UI'en.

Dette er primært fordi der skal oprettes de forskellige knapper som brugeren kan trykke på for at udføre de forskellige use cases. Da GWT er udviklet på en måde, så vores clickhandlers skal defineres umiddelbart efter en knap er blevet defineret, smelter vores controllers sammen med UI'en. Dette gør at vi får et datalag og et controller / UI lag.

Test

I forbindelse med udviklingen af GWT webapplikationen, har vi foretaget test for at se om vores program virkede og for at overholde Unified Proces med testning på hver iteration.

I dette projekt er GWT'en mest af alt brugerorienteret, og det er derfor ikke mulig for os at lave en automatiseret test af selve UI'en. Derfor bliver vi nødt til at teste de forskellige metoder manuelt, og efterlade billeder og tekst der viser, at vores program virker som det skal.

Den anden del af testen bliver test af selve vores database. Dette kan let laves automatiseret, men det skal nævnes at vi ikke har oprettet en 'slet' funktion i vores projekt. Dette betyder, at hvis man udfører testen 2 gange i træk, vil den fejle. Dette er fordi vores test indsætter et element efter at have undersøgt at et element eksistere, og sletter man ikke dette element manuelt, fejler testen. Derfor skal man slette dét element der bliver indsat i testen inden testen begynder. Vi starter med at teste GWT'en:

Test - GWT DAO

Vi starter med at se om login fungerer. For at gøre dette, forsøge det at logge ind som superbrugeren med ID 1 og password 'PassTE10'. Derefter forsøges igen, bare hvor der skrives koden forkert ('PassTE11'). Til sidst logges korrekt ind som en værkfører (ID 3, password passGJ03). Hvis der stadig opnås samme funktioner som en superbruger når der logges ind, er vores login ikke integreret ordentligt. Proceduren var således:

1. Log ind som superbruger med korrekt login: bilag 1
2. Log ind som superbruger med forket login: bilag 2
3. Log ind som værkføre med korrekt login bilag 3

Nu da dette virker, vil vi gerne vise vores use cases. For at spare plads vil vi gerne nøjes med at dokumentere UC1 igennem, da resten af use cases'ne er meget ens implementeret. Vi vil gøre følgende:

1. For at teste at vis og opret fungerer, vil vi:
    a. Kigge listen igennem med brugere: Bilag 4
    b. Indsætte en bruger vi ved ikke eksistere: Bilag 4
    c. kigge listen igennem igen: Bilag 4

2. For at teste om ret og slet fungerer, vil vi:
    a. Kigge listen igennem med brugere: Bilag 5
    b. Slette en aktiv bruger: Bilag 5
    c. Rette denne brugers navn: Bilag 5

d. Kigge listen med inaktive brugere igennem og se, om den er opdateret korrekt: Bilag 6

I og med at vores implementation er meget ens, har vi ikke lyst til at vise flere af disse test, da de fylder for meget, og vi har testen programmet igennem grundigt. Dette er blot en dokumentation for én af dem og resten af use cases er brugertestet fuldt ud og fungerer, de medtages blot ikke i denne rapport da de ville fylde for meget.

J-Unit test DAO

Den anden del af testen består i at lave Junit tests på DAO'en. I dette projekt består de fleste database kommandoer mest af get og create metoder, og den eneste der ejer andre funktioner end disse, er brugeradministration. Derfor vil vi nøjes med at teste denne grundigt, da andre test er mere eller mindre identisk til den vi foretager her.

Vi forsøger at teste alle dens funktioner, hvilket er get, create, update og delete. Vores test fungere således:

- Gem alle navnene på de forskellige bruger i en liste
- Indsæt en aktiv bruger
- Gem den nye liste, og sammenlign med den gamle. Er de ikke ens, må både get og create virke
- Slet brugeren der lige er oprettet
- Undersøg om brugeren er aktiv eller ej. Er han ikke aktiv, må delete virke.
- Ændrer navnet på brugeren, og gem navnet i en variabel
- Er brugerens navn det samme som variablens, er testen lykkedes

Lykkes alt dette, virker vores datalag på brugeradministration. Det skal nævnes, at denne test fejler hvis enten brugeren man indsætter ikke står på en bestemt plads, eller hvis man ikke manuelt sletter brugeren efter hver test fra databasen.

Afgrænsning

Alle projekter oplever at dele af den forventede funktioner ikke altid bliver implimenteret fuldt ud. Dette har også gjort sig gældende for vores projekt, da vi pga. begrænset tid, har en række uforløst funktioner der ikke er blevet udarbejdet komplet endnu. Følgende er:

- Vores fejlbeskeder er ikke særlig brugervenlige. Vi sender fejlbeskeder fra SQL til brugerfladen, og man skal ofte have en viden om database for at kunne forstå fejlbeskeden. Vi kan ikke forvente at pharmaceuter har denne viden.

- På nuværende tidspunkt er det muligt for superbrugeren at rette hans eget niveau. Denne bruger har derefter stadig superbruger status indtil personen logger ud igen, og kan derfor udfører ting personen ikke burde.

- Systemet tager på nuværende tidspunkt ikke højde for scenariet hvor en bruger med præcis det samme navn, ini, password, active og niveau oprettes. En brugers primærnøgle er autogenereret, og den vil blot lave endnu en identisk bruger, bare med anderledes ID - dog er "cpr" attributen unik og således kan man ikke have to brugere i systemet med samme CPR nummer.
-

Delkonklusion GWT

GWT har vist sig at være et fremragende værktøj til, forholdsvist let, at lave hjemmesider. Vi har fået implementeret alt det nødvendige for at man kanne navigere sig fornuftig rundt på siden, og de forskellige brugere får kun de muligheder præsenteret som det er meningen de skal have, i henhold til de usecases der er givet i oplægget.

Et sted GWT har givet os problemer er specielt ved testningen af den. Det er utrolig svært at dokumentere ordentligt, at de forskellige knapper gør hvad de skal, og der er meget manuelt arbejde i at teste vores Junit test, da man selv skal fjerne dét element der bliver indsat.

Dette er dog kun en prototype, så der er naturligvis visse mangler der ville være udfyldt i et færdigt produkt. Vores login er let at manipulere med, og vores fejlbeskeder vi sender til brugeren kræver viden om SQL, og dét kan vi ikke forvente at en "almindelig" bruger har kendskab til.

Konklusion

Vi har fået gennemført projektperioden med et godt udgangspunkt og en funktionsdygtig prototype. Undervejs har vi fulgt udviklingsmetoden Unified Proces og udført tests løbende.

Rapporten dokumenterer hvordan alle komponenter i systemet tilsammen opfylder de opstillede uses cases fra CDIO final oplægget. Som tidligere nævnt bygger systemet på gamle iterationer af CDIO projekter, men det er først ved dette CDIO final at komponenter for alvor bliver testet sammen. Dette har givet en del problemer og vi har måttet dedikere en del tid til at forbedre og udvikle funktioner.

Vægt simulatoren har under 3-ugers perioden undergået store forandringer både ift. generelle rettelser og til en bedre implementering af RM 20 kommandoen. Vægtsimulatoren blev prioriteret igen under dette projekt for at sikrer at vores test med simulatoren ville afspejle systemets funktionalitet ift. den fysiske vægt.

I forlængelse af at teste systemet prioriterede vi også gøre ASE'en funktionsdygtig. Allerede ved første gennemtestning af den fysisk vægt virkede systemet. Dette kan i høj grad tilskrives den arbejdsfordeling hvor ASE'en og vægtsimulatren blev prioriteret. Vi har fået udviklet en ASE der lever op til de opstillede krav for den og således har vi en god prototype.

GWT delen af systemet byggede også på tidligere version fra CDIO 3. I projektet brugte vi denne tidligere implementation struktur og udbyggede funktionaliteten til også at inddrage farmaceut og værkfører. Vi fik yderligere gjort GUI'en specifik for den enkelte bruger af systemet. F.eks kunne værkførern kun interagere og se den interface han var sikkerhedsgodkendt til af systemet.

Overordnet set opfylder systemet kravene fra CDIO final oplægget, for så vidt muligt. Projektet bær stadig prædikatet prototype, men vi har forsøgt at prioritere hvilke arbejdsopgaver der var vigtigst for os at opnå og så ellers erkendt vores og projektets begrænsninger.Disse begrænsninger er blevet skitseret i de respektive afgrænsning afsnit.

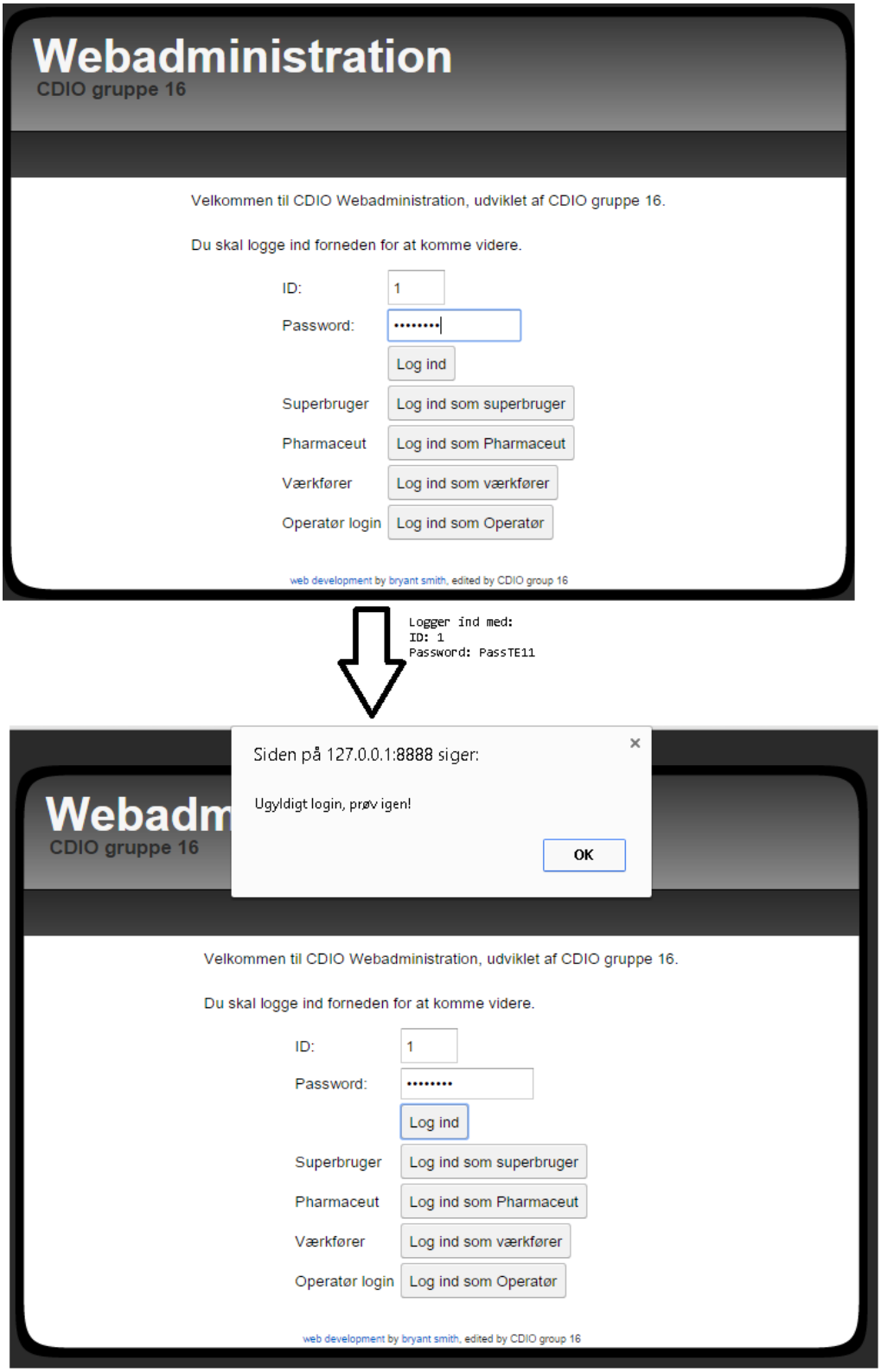
Alt i alt er projektperioden forløbet rigtig godt, der har været et suverænt samarbejde i gruppen og vi har været rigtig gode til at fordele arbejdsbyrden ud på alle mand således at projektet blev mere overskueligt for hvert enkelt individ.

Bilag

Bilag 1 - Log ind

```
Logger ind med:
ID: 1
Password: PassTE10
```

Bilag 2 - Log ind

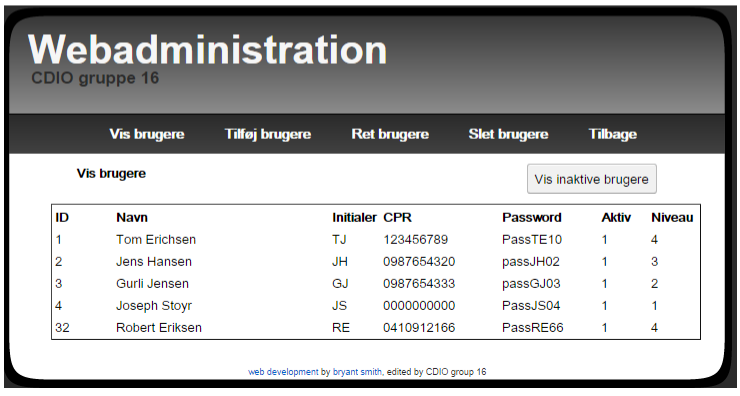


Bilag 3 - Login

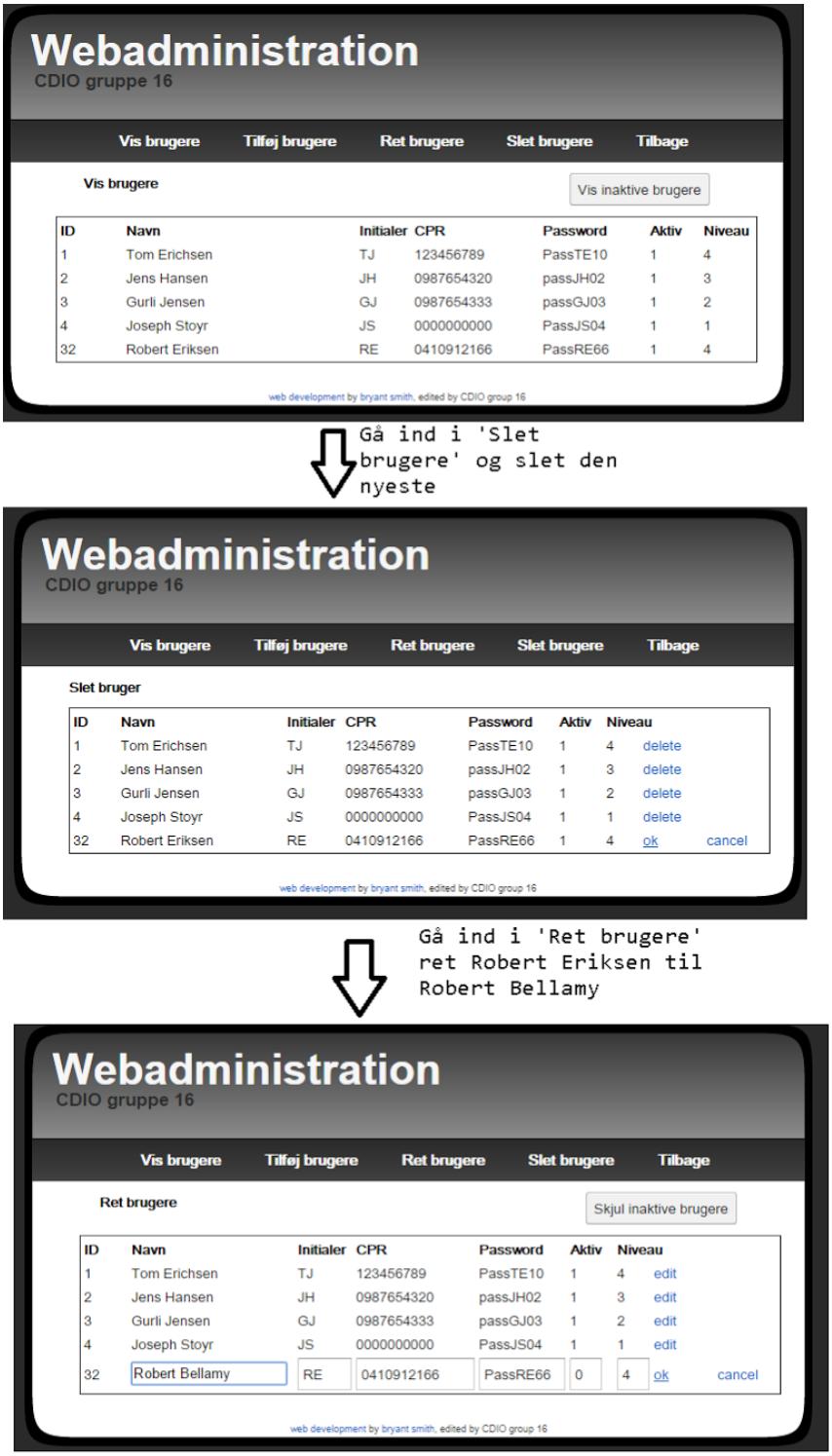
Bilag 4 - Tilføj bruger

Gå ind i 'tilføj bruger'
og opret en ny

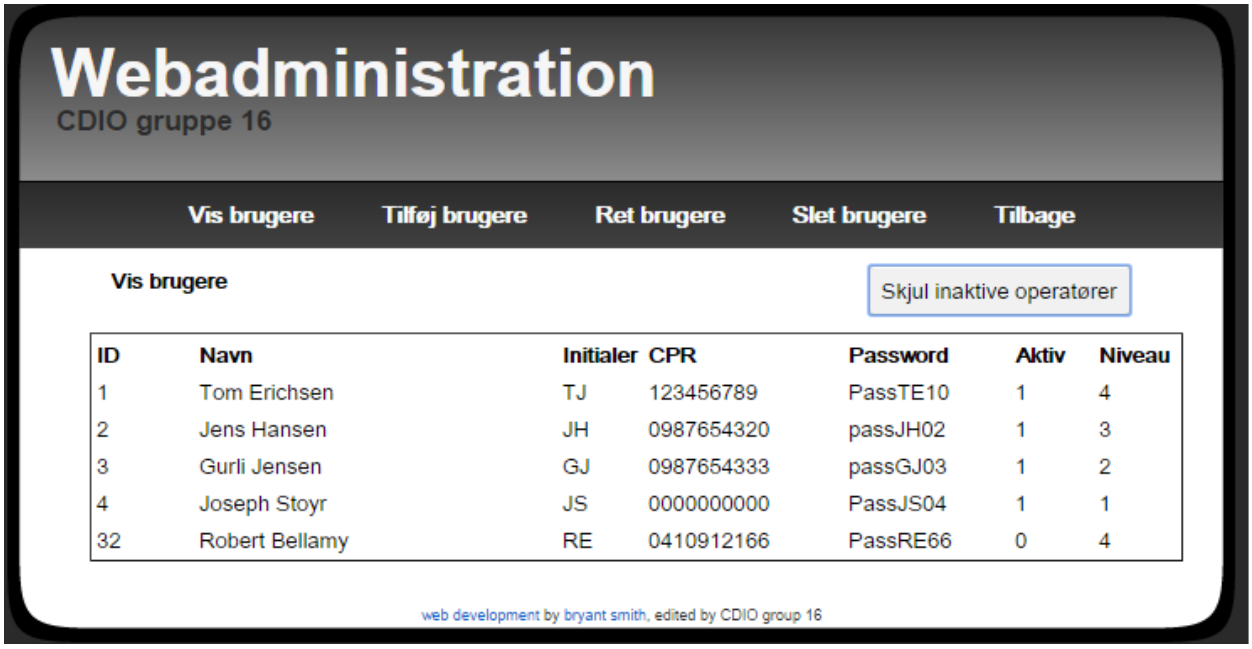
Tryk 'Tilføj' og gå
ind i 'Vis bruger'



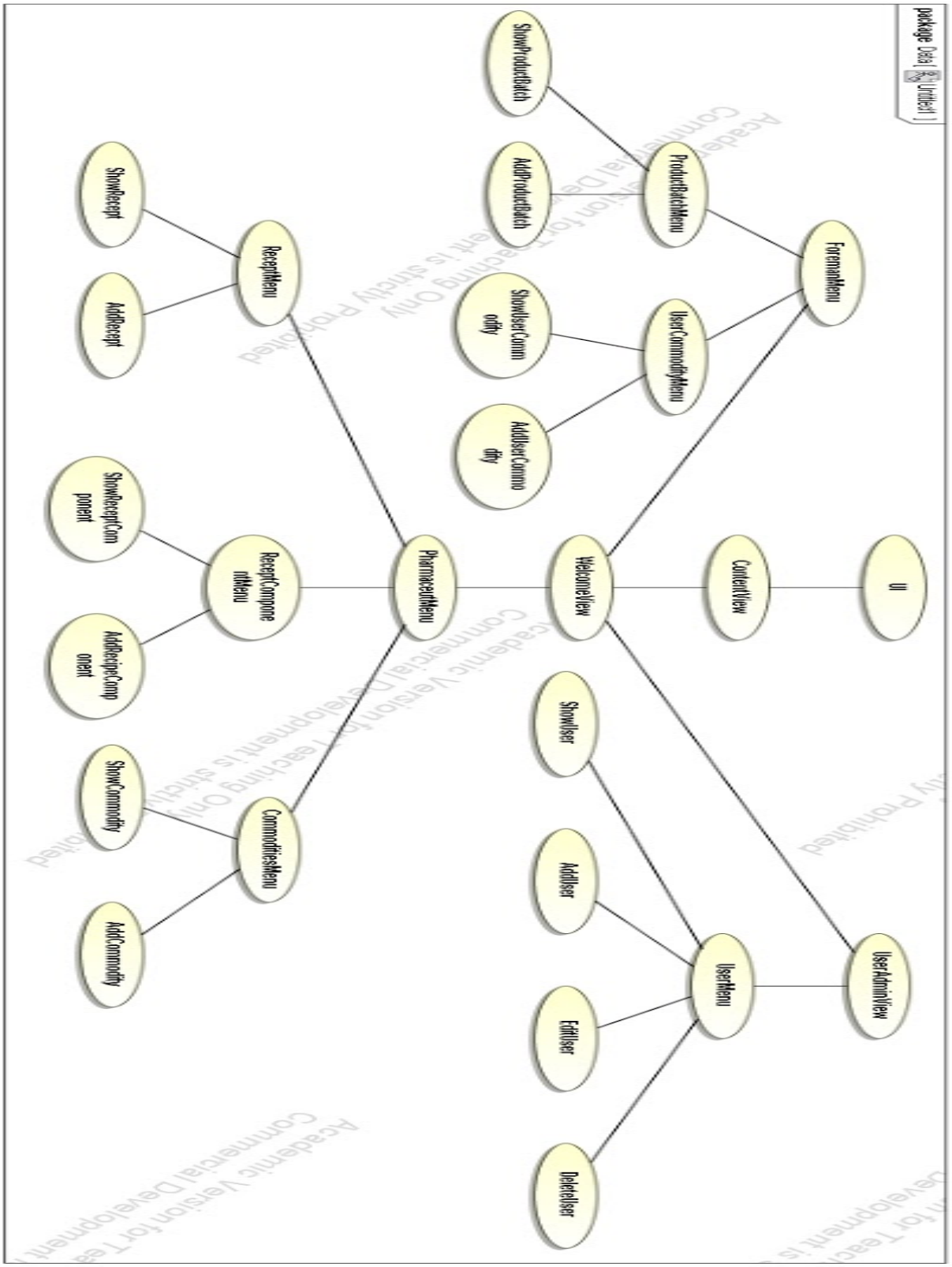
Bilag 5 - Ret bruger



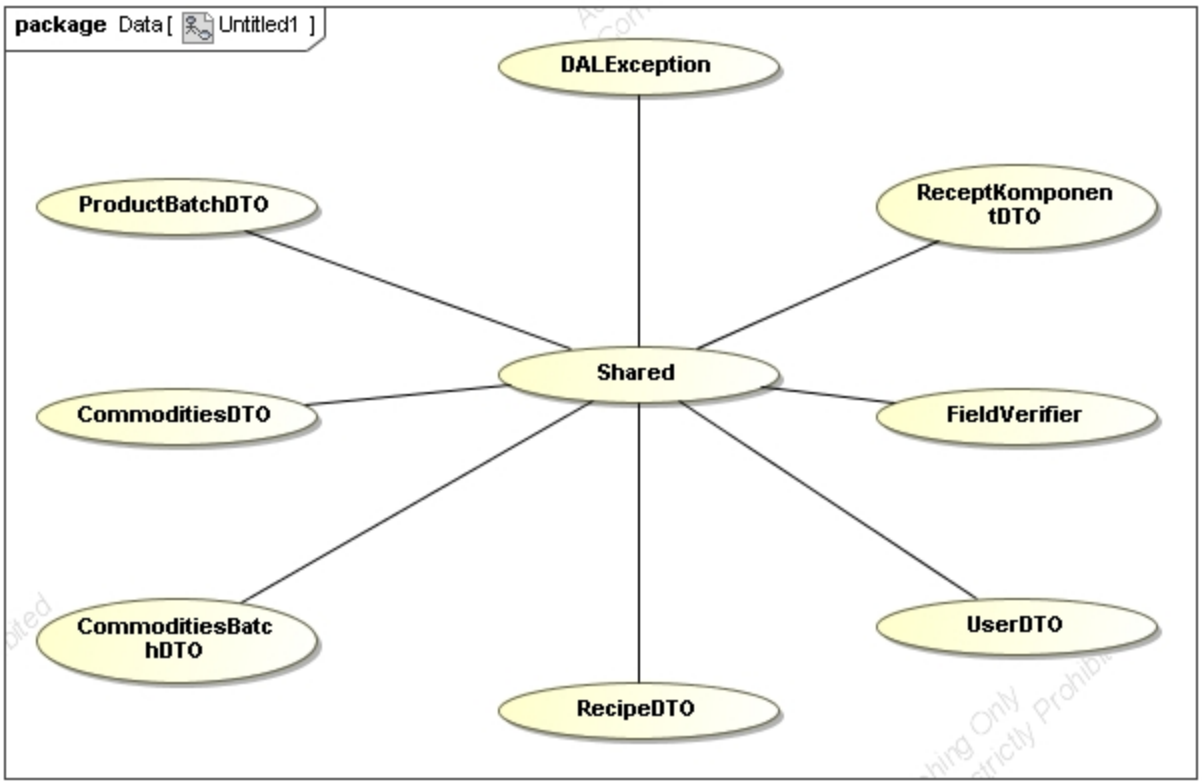
Bilag 6 - Ret bruger vis



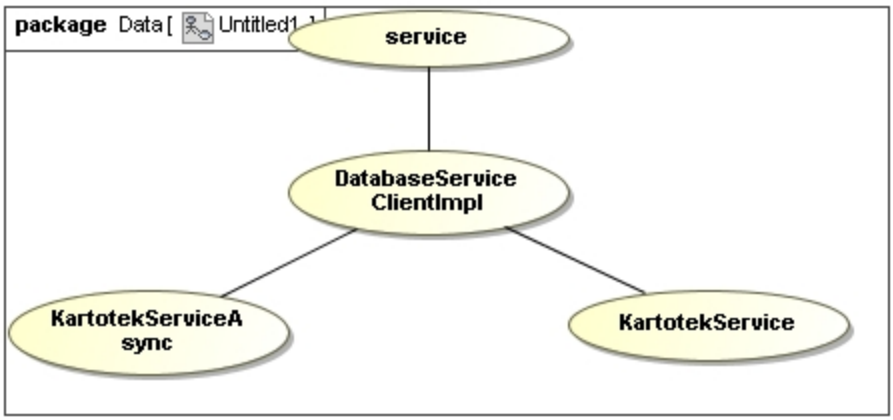
Bilag 7 - Klasse oversigt



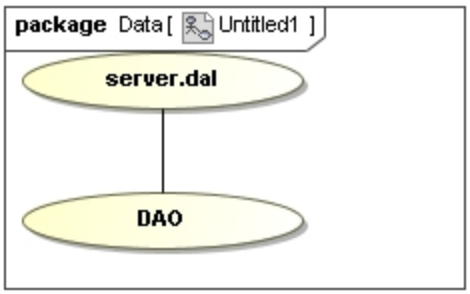
Bilag 8 - Shared package



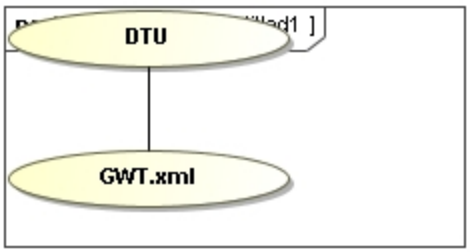
Bilag 9 - Service package



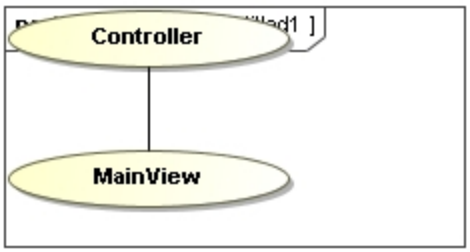
Bilag 10



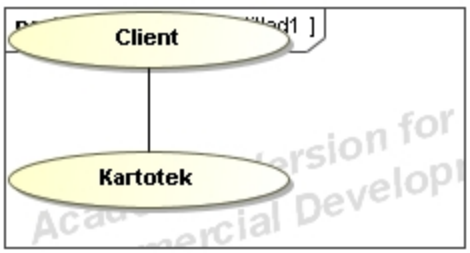
Bilag 11



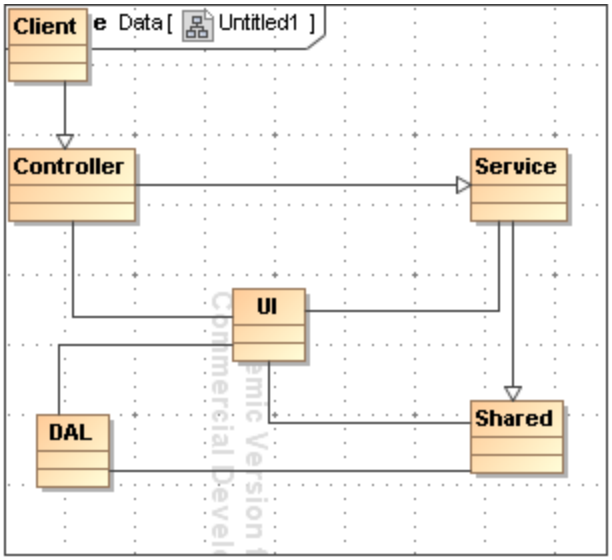
Bilag 12



Bilag 13



Bilag 14



Bilag 15 - Use case beskrivelser

Bruger administration UC1.1	
Use Case beskrivelse	aGennemfør : Administratoren kan logge sig ind i systemet for at oprette en bruger.
Aktør	Administrator aktøren
Interessenter	Farmaceut, Værkfører og Operatør.
Forudsætninger	Administrator eksisterer i forvejen.
Succes	Brugeren er oprettet
Main Succes Scenario	Administratoren logger på sin side med sit ID og password. Administratoren opretter en bruger Brugeren oprettes Brugerens ID , navn, initialer, password samt brugerens rolle angives Administratoren logger ud.

Alternative scenarier	Administratoren logger på sin side med sit ID og password. Administratoren opretter en bruger. Fejlmeddelelser kommer frem. Brugeren logger ud ..
Teknologi og data Variationer List	Java – DB- SQL – GWT - 3 lags modellen
Frequency of Use	En gang imellem

Bruger administration UC1.2	
Use Case beskrivelse	aGennemfør : Administratoren kan logge sig ind i systemet for at rette en bruger.
Aktør	Administrator aktøren
Interessenter	Farmaceut, Værkfører og Operatør.
Forudsætninger	Administrator eksisterer i forvejen.
Succes	Brugeren er rettet
Main Succes Scenario	1. Administratoren logger på sin side med sit ID og password. 2. Administratoren retter en bruger 3. Brugeren rettes 4. Rettelsen opdateres 5. Administratoren logger ud.
Alternative scenarier	1. Administratoren logger på sin side med sit ID og password. 2. Administratoren retter en bruger. 3. Fejlmeddelelser kommer frem. 4. Brugeren logger ud ..
Teknologi og data Variationer List	Java – DB- SQL – GWT - 3 lags modellen
Frequency of Use	Sjælden

Bruger administration UC1.3	
Use Case beskrivelse	aGennemfør : Administratoren kan logge sig ind i systemet for at slette en bruger.
Aktør	Administrator aktøren
Interessenter	Farmaceut, Værkfører og Operatør.
Forudsætninger	Administrator eksisterer i forvejen.
Succes	Brugeren er slettet
Main Succes Scenario	1. Administratoren logger på sin side med sit ID og password. 2. Administratoren slette en bruger 3. Brugeren slettes 4. Administratoren logger ud.
Alternative scenarier	1. Administratoren logger på sin side med sit ID og password. 2. Administratoren sletter en bruger. 3. Fejlmeddelelser kommer frem. a. En bruger med rollen Operatør der én gang er oprettet i systemet kan ikke slettes. 4. Brugeren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	Sjælden

Bruger administration UC1.4	
Use Case beskrivelse	Gennemfør : Administratoren kan logge sig ind i systemet for at vise en bruger.
Aktør	Administrator aktøren

Interessenter	Farmaceut, Værkfører og Operatør.
Forudsætninger	Administrator eksisterer i forvejen.
Succes	Brugeren er vist
Main Succes Scenario	1. Administratoren logger på sin side med sit ID og password. 2. Administratoren viser en bruger 3. Brugeren vises 4. Administratoren logger ud.
Alternative scenarier	Administratoren logger på sin side med sit ID og password. Administratoren viser en bruger. Fejlmeddelelser kommer frem. Brugeren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	Sker tit

Råvare administration UC2.1	
Use Case beskrivelse	Gennemfør : farmaceut aktøren kan logge sig ind i systemet for at oprette råvarer. Råvare defineres ved et råvareNr, navn samt leverandør.
Aktør	Farmaceut aktøren
Interessenter	Farmaceut, Værkfører , Operatør og leverandør
Forudsætninger	Farmaceuten eksisterer i forvejen.
Succes	Råvarer er oprettet

Main Succes Scenario	1. Farmaceuten logger på sin side med sit ID og password. 2. Farmaceuten opretter råvarer 3. råvarer oprettes 4. Farmaceuten logger ud.
Alternative scenarier	1. Farmaceuten logger på sin side med sit ID og password. 2. Farmaceuten opretter råvarer. 3. Fejlmeddelelser kommer frem. 4. Brugeren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	Sker tit

Råvare administration UC2.2	
Use Case beskrivelse	Gennemfør : farmaceut aktøren kan logge sig ind i systemet for at rette råvarer.
Aktør	Farmaceut aktøren
Interessenter	Farmaceut, Værkfører , Operatør og leverandør
Forudsætninger	Farmaceuten eksisterer i forvejen.
Succes	Råvarer er rettet
Main Succes Scenario	1. Farmaceuten logger på sin side med sit ID og password. 2. Farmaceuten retter råvarer 3. råvarer rettes 4. Følgende opdateres: råvareNr, navn samt leverandør 5. Farmaceuten logger ud.

Alternative scenarier	1. Farmaceuten logger på sin side med sit ID og password. 2. Farmaceuten retter råvarer. 3. Fejlmeddelelser kommer frem. 4. Brugeren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	En gang imellem

Råvare administration UC2.3	
Use Case beskrivelse	Gennemfør : farmaceut aktøren kan logge sig ind i systemet for at vise råvarer.
Aktør	Farmaceut aktøren
Interessenter	Farmaceut, Værkfører , Operatør og leverandør
Forudsætninger	Farmaceuten eksisterer i forvejen.
Succes	Råvarer er vist
Main Succes Scenario	1. Farmaceuten logger på sin side med sit ID og password. 2. Farmaceuten viser råvarer 3. råvarer vises 4. Farmaceuten logger ud.
Alternative scenarier	5. Farmaceuten logger på sin side med sit ID og password. 6. Farmaceuten viser råvarer. 7. Fejlmeddelelser kommer frem. 8. Brugeren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	altid

Recept administration UC3.1	
Use Case beskrivelse	Gennemfør : farmaceut aktøren kan logge sig ind i systemet for at oprette recepter. En recept defineres ved et receptNr, navn samt en sekvens af receptkomponenter. En ***receptkomponent*** består af en råvare (type), en mængde samt en tolerance.
Aktør	Farmaceut aktøren
Interessenter	Farmaceut, Værkfører , Operatør og leverandør
Forudsætninger	Farmaceuten eksisterer i forvejen.
Succes	Råvarer er vist
Main Succes Scenario	1. Farmaceuten logger på sin side med sit ID og password. 2. Farmaceuten opretter en recept. 3. en recept oprettes. 4. Farmaceuten logger ud.
Alternative scenarier	9. Farmaceuten logger på sin side med sit ID og password. 10. Farmaceuten oprette en recept. 11. Fejlmeddelelser kommer frem. 12. Brugeren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	Altid

Recept administration UC3.2

Use Case beskrivelse	Gennemfør : farmaceut aktøren kan logge sig ind i systemet for at vise recepter.
Aktør	Farmaceut aktøren
Interessenter	Farmaceut, Værkfører , Operatør og leverandør
Forudsætninger	Farmaceuten eksisterer i forvejen.
Succes	Råvarer er vist
Main Succes Scenario	Farmaceuten logger på sin side med sit ID og password. Farmaceuten at vise recepter. recepter vises Farmaceuten logger ud.
Alternative scenarier	1. Farmaceuten logger på sin side med sit ID og password. 2. Farmaceuten viser recepter. 3. Fejlmeddelelser kommer frem. 4. Brugeren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	Altid

Råvarebatch administration UC4.1	
Use Case beskrivelse	Gennemfør : værkfører aktøren kan logge sig ind i systemet for at oprette råvarebatches . En råvarebatch defineres ved et råvarebatchNr samt mængde
Aktør	Værkfører aktøren

Interessenter	Farmaceut, Værkfører , Operatør og leverandør
Forudsætninger	Værkføreren eksisterer i forvejen.
Succes	Råvarebatches er oprettet.
Main Succes Scenario	1. Værkføreren logger på sin side med sit ID og password. 2. Værkføreren opretter råvarebatches . 3. Råvarebatches oprettes. 4. Værkføreren logger ud.
Alternative scenarier	1. Værkføreren logger på sin side med sit ID og password. 2. Værkføreren opretter råvarebatches . 3. Fejlmeddelelser kommer frem. 4. Værkføreren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	Altid

Råvarebatch administration UC4.2	
Use Case beskrivelse	Gennemfør : værkfører aktøren kan logge sig ind i systemet for at vise råvarebatches .
Aktør	Værkfører aktøren
Interessenter	Farmaceut, Værkfører , Operatør og leverandør
Forudsætninger	Værkføreren eksisterer i forvejen.
Succes	Råvarebatches er vist.

Main Succes Scenario	1. Værkføreren logger på sin side med sit ID og password. 2. Værkføreren viser råvarebatches . . 3. Værkføreren logger ud.
Alternative scenarier	1. Værkføreren logger på sin side med sit ID og password. 2. Værkføreren viser recepter. 3. Fejlmeddelelser kommer frem. 4. Værkføreren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	Altid

Produktbatch administration UC5.1	
Use Case beskrivelse	Gennemfør : værkfører aktøren kan logge sig ind i systemet for at oprette produktbatches. En *produktbatch* defineres ved et produktbatchNr (entydigt), nummeret på den *recept produktbatchen* skal produceres udfra, dato for oprettelse, samt oplysning om status for batchen. Status kan være: oprettet / under produktion / afsluttet.
Aktør	Værkfører aktøren
Interessenter	Farmaceut, Værkfører , Operatør og leverandør
Forudsætninger	Værkføreren eksisterer i forvejen.
Succes	Produktbatches er oprettet.

Main Succes Scenario	1. Værkføreren logger på sin side med sit ID og password. 2. Værkføreren opretter Produktbatches. 3. Produktbatches udprintes og uddeles til en udvalgt operatør 4. Værkføreren logger ud.
Alternative scenarier	1. Værkføreren logger på sin side med sit ID og password. 2. Værkføreren opretter Produktbatches. 3. Fejlmeddelelser kommer frem. 4. Værkføreren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	Altid

Produktbatch administration UC5.2	
Use Case beskrivelse	Gennemfør : værkfører aktøren kan logge sig ind i systemet for at vise produktbatches. En *produktbatch* defineres ved et produktbatchNr (entydigt), nummeret på den *recept produktbatchen* skal produceres udfra, dato for oprettelse, samt oplysning om status for batchen. Status kan være: oprettet / under produktion / afsluttet.
Aktør	Værkfører aktøren
Interessenter	Farmaceut, Værkfører , Operatør og leverandør
Forudsætninger	Værkføreren eksisterer i forvejen.
Succes	Produktbatches er vist.

Main Succes Scenario	1. Værkføreren logger på sin side med sit ID og password. 2. Værkføreren viser Produktbatches. 3. Produktbatches vises 4. Værkføreren logger ud.
Alternative scenarier	1. Værkføreren logger på sin side med sit ID og password. 2. Værkføreren viser Produktbatches. 3. Fejlmeddelelser kommer frem. 4. Værkføreren logger ud ..
Teknologi og data Variationer List	Java – DB - SQL – GWT - 3 lags modellen
Frequency of Use	Altid

Bilag 16 - GWT kildekode

```java
package dtu.client.ui;

import com.google.gwt.event.dom.client.ClickEvent;

public class AdministratorMenu extends Composite {
    private HorizontalPanel hPanel = new HorizontalPanel();

    // receive reference to MainView for call back
    public AdministratorMenu(final MainView main, final int level) {
        initWidget(this.hPanel);

        Anchor bruger_Administration = new Anchor("Bruger");
        hPanel.add(bruger_Administration);
        bruger_Administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showUserMenu(level);
            }
        });

        Anchor raavare_administration = new Anchor("Råvare");
        hPanel.add(raavare_administration);
        raavare_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showCommoditiesMenu(level);
            }
        });

        Anchor raavarebatch_administration = new Anchor("Råvarebatch");
        hPanel.add(raavarebatch_administration);
        raavarebatch_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showCommoditiesBatchMenu(level);
            }
        });

        Anchor recept_administration = new Anchor("Recept");
        hPanel.add(recept_administration);
        recept_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showRecipeMenu(level);
            }
        });

        Anchor receptKomponent_administration = new Anchor("Receptkomponent");
        hPanel.add(receptKomponent_administration);
        receptKomponent_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showRecipeKomponentMenu(level);
            }
        });

        Anchor produktbatch_administration = new Anchor("Produktbatch");
        hPanel.add(produktbatch_administration);
        produktbatch_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showProductBatchMenu(level);
            }
        });
    }
}

```

```java
package dtu.client.ui;


import com.google.gwt.user.client.Window;

public class CommoditiesAddView extends Composite {
    VerticalPanel addPanel;

    // controls
    Label idLbl;
    Label nameLbl;
    Label lvrLbl;

    TextBox idTxt;
    TextBox nameTxt;
    TextBox lvrTxt;

    Button save = new Button("Tilf\u00F8j råvare");

    // valid fields
    boolean idValid = false;
    boolean nameValid = false;
    boolean lvrValid = false;

    public CommoditiesAddView(final DatabaseServiceClientImpl clientImpl) {

        addPanel = new VerticalPanel();
        initWidget(this.addPanel);

        FlexTable addTable = new FlexTable();

        Label pageTitleLbl = new Label("Tilføj råvare");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.addStyleName("spacing-vertical");
        addPanel.add(pageTitleLbl);

        idLbl = new Label("Råvare ID:");
        idTxt = new TextBox();
        Label idRulesLbl = new Label("(Heltal kun)");
        addTable.setWidget(0, 0, idLbl);
        addTable.setWidget(0, 1, idTxt);
        addTable.setWidget(0, 2, idRulesLbl);

        nameLbl = new Label("Råvare navn:");
        nameTxt = new TextBox();
        Label nameRulesLbl = new Label("(minimum 2 tegn og maksimum 20)");
        addTable.setWidget(3, 0, nameLbl);
        addTable.setWidget(3, 1, nameTxt);
        addTable.setWidget(3, 2, nameRulesLbl);

        lvrLbl = new Label("Leverandør:");
        lvrTxt = new TextBox();
        Label lvrRulesLbl = new Label("(minimum 2 og maksimum 20 tegn)");
        addTable.setWidget(5, 0, lvrLbl);
        addTable.setWidget(5, 1, lvrTxt);
        addTable.setWidget(5, 2, lvrRulesLbl);

        idTxt.setStyleName("gwt-TextBox-invalidEntry");
        nameTxt.setStyleName("gwt-TextBox-invalidEntry");
        lvrTxt.setStyleName("gwt-TextBox-invalidEntry");

        // use unicode escape sequence \u00F8 for '\u00F8'
```

```
80          save = new Button("Tilføj råvare");
81          save.setEnabled(false);
82          addTable.setWidget(6, 1, save);
83
84          save.addClickHandler(new ClickHandler() {
85
86              @Override
87              public void onClick(ClickEvent event) {
88
89                  // create new RaavareDTO
90                  CommoditiesDTO newRaavare = new
    CommoditiesDTO(Integer.valueOf(idTxt.getText()), nameTxt.getText(),
    lvrTxt.getText());
91                  // save on server
92                  clientImpl.service.createCommodity(newRaavare, new
    AsyncCallback<Void>() {
93
94                      @Override
95                      public void onSuccess(Void result) {
96                          Window.alert("Råvare gemt i database.");
97                      }
98
99                      @Override
100                     public void onFailure(Throwable caught) {
101                         Window.alert("Server fejl!" + caught.getMessage());
102                     }
103
104                 });
105             }
106         });
107
108         idTxt.addKeyUpHandler(new KeyUpHandler(){
109
110             @Override
111             public void onKeyUp(KeyUpEvent event) {
112                 if (!FieldVerifier.isValidID(idTxt.getText())) {
113                     idTxt.setStyleName("gwt-TextBox-invalidEntry");
114                     idValid = false;
115                 }
116                 else {
117                     idTxt.removeStyleName("gwt-TextBox-invalidEntry");
118                     idValid = true;
119                 }
120                 checkFormValid();
121             }
122
123         });
124
125         nameTxt.addKeyUpHandler(new KeyUpHandler(){
126
127             @Override
128             public void onKeyUp(KeyUpEvent event) {
129                 if (!FieldVerifier.isValidName(nameTxt.getText())) {
130                     nameTxt.setStyleName("gwt-TextBox-invalidEntry");
131                     nameValid = false;
132                 }
133                 else {
134                     nameTxt.removeStyleName("gwt-TextBox-invalidEntry");
135                     nameValid = true;
136                 }
137                 checkFormValid();
138             }
```

```
139
140         });
141
142         lvrTxt.addKeyUpHandler(new KeyUpHandler(){
143
144             @Override
145             public void onKeyUp(KeyUpEvent event) {
146                 if (!FieldVerifier.isValidName(lvrTxt.getText())) {
147                     lvrTxt.setStyleName("gwt-TextBox-invalidEntry");
148                     lvrValid = false;
149                 }
150                 else {
151                     lvrTxt.removeStyleName("gwt-TextBox-invalidEntry");
152                     lvrValid = true;
153                 }
154                 checkFormValid();
155             }
156
157         });
158         addPanel.add(addTable);
159     }
160
161     private void checkFormValid() {
162         if (idValid&&nameValid&&lvrValid)
163             save.setEnabled(true);
164         else
165             save.setEnabled(false);
166     }
167
168 }
169
```

```java
package dtu.client.ui;


import com.google.gwt.user.client.Window;

public class CommoditiesBatchAddView extends Composite {
    VerticalPanel addPanel;

    // controls
    Label raavareBatchIdLbl;
    Label raavareidLbl;
    Label maengdeLbl;

    TextBox raavareBatchIdTxt;
    TextBox raavareidTxt;
    TextBox maengdeTxt;

    Button save = new Button("Tilf\u00F8j råvarebatch");

    // valid fields
    boolean raav_BatIDValid = false;
    boolean raavIDValid = false;
    boolean maengdeValid = false;

    public CommoditiesBatchAddView(final DatabaseServiceClientImpl clientImpl) {

        addPanel = new VerticalPanel();
        initWidget(this.addPanel);

        FlexTable addTable = new FlexTable();

        Label pageTitleLbl = new Label("Tilføj råvarebatch ");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.addStyleName("spacing-vertical");
        addPanel.add(pageTitleLbl);

        raavareBatchIdLbl = new Label("Råvarebatch ID:");
        raavareBatchIdTxt = new TextBox();
        Label rbIDRulesLbl = new Label("(Heltal kun)");
        addTable.setWidget(0, 0, raavareBatchIdLbl);
        addTable.setWidget(0, 1, raavareBatchIdTxt);
        addTable.setWidget(0, 2, rbIDRulesLbl);

        raavareidLbl = new Label("Råvare ID:");
        raavareidTxt = new TextBox();
        Label rvIDRulesLbl = new Label("(Heltal kun, skal findes i forvejen)");
        addTable.setWidget(1, 0, raavareidLbl);
        addTable.setWidget(1, 1, raavareidTxt);
        addTable.setWidget(1, 2, rvIDRulesLbl);

        maengdeLbl = new Label("Mængde:");
        maengdeTxt = new TextBox();
        Label maengdeRulesLbl = new Label("(Decimaltal)");
        addTable.setWidget(2, 0, maengdeLbl);
        addTable.setWidget(2, 1, maengdeTxt);
        addTable.setWidget(2, 2, maengdeRulesLbl);

        raavareBatchIdTxt.setStyleName("gwt-TextBox-invalidEntry");
        raavareidLbl.setStyleName("gwt-TextBox-invalidEntry");
        maengdeTxt.setStyleName("gwt-TextBox-invalidEntry");

        // use unicode escape sequence \u00F8 for '◆'
```

```
 82         save.setEnabled(false);
 83         addTable.setWidget(5, 1, save);
 84 
 85         save.addClickHandler(new ClickHandler() {
 86 
 87             @Override
 88             public void onClick(ClickEvent event) {
 89 
 90                 CommoditiesBatchDTO newRaavareBatch = new
    CommoditiesBatchDTO(Integer.valueOf(raavareBatchIdTxt.getText()),
    Integer.valueOf(raavareidTxt.getText()), Double.valueOf(maengdeTxt.getText()));
 91 
 92                 // save on server
 93                 clientImpl.service.createCommodityBatch(newRaavareBatch, new
    AsyncCallback<Void>() {
 94 
 95                     @Override
 96                     public void onSuccess(Void result) {
 97                         Window.alert("Råvarebatch gemt i database.");
 98                     }
 99 
100                     @Override
101                     public void onFailure(Throwable caught) {
102                         Window.alert("Server fejl!" + caught.getMessage());
103                     }
104 
105                 });
106             }
107         });
108 
109         raavareBatchIdTxt.addKeyUpHandler(new KeyUpHandler(){
110 
111             @Override
112             public void onKeyUp(KeyUpEvent event) {
113                 if (!FieldVerifier.isValidID(raavareBatchIdTxt.getText())) {
114                     raavareBatchIdTxt.setStyleName("gwt-TextBox-invalidEntry");
115                     raav_BatIDValid = false;
116                 }
117                 else {
118                     raavareBatchIdTxt.removeStyleName("gwt-TextBox-invalidEntry");
119                     raav_BatIDValid = true;
120                 }
121                 checkFormValid();
122             }
123 
124         });
125 
126         raavareidTxt.addKeyUpHandler(new KeyUpHandler(){
127 
128             @Override
129             public void onKeyUp(KeyUpEvent event) {
130                 if (!FieldVerifier.isValidID(raavareidTxt.getText())) {
131                     raavareidTxt.setStyleName("gwt-TextBox-invalidEntry");
132                     raavIDValid = false;
133                 }
134                 else {
135                     raavareidTxt.removeStyleName("gwt-TextBox-invalidEntry");
136                     raavIDValid = true;
137                 }
138                 checkFormValid();
139             }
140 
```

```
141         });
142
143         maengdeTxt.addKeyUpHandler(new KeyUpHandler(){
144
145             @Override
146             public void onKeyUp(KeyUpEvent event) {
147                 if (!FieldVerifier.IsValidMaengde(maengdeTxt.getText())) {
148                     maengdeTxt.setStyleName("gwt-TextBox-invalidEntry");
149                     maengdeValid = false;
150                 }
151                 else {
152                     maengdeTxt.removeStyleName("gwt-TextBox-invalidEntry");
153                     maengdeValid = true;
154                 }
155                 checkFormValid();
156             }
157
158         });
159         addPanel.add(addTable);
160     }
161
162     private void checkFormValid() {
163         if (raavIDValid&&maengdeValid&&raav_BatIDValid)
164             save.setEnabled(true);
165         else
166             save.setEnabled(false);
167     }
168
169 }
170
```

```java
package dtu.shared;

import java.io.Serializable;

/**

 * Operatør Data Transfer Object

 */

public class CommoditiesBatchDTO implements Serializable {

    /** Operatør id i området 1-99999999. Vælges af brugerne */
    int rbId;

    /** Operatør navn min. 2 max. 20 karakterer */
    int raavareId;

    /** Operatør cpr-nr 10 karakterer */
    double maengde;

    /** Operatør password min. 7 max. 8 karakterer */
//  double nomNetto;

    /** Operatør aktiv (1) eller inaktiv (0) */
//  double tolerance;

    public CommoditiesBatchDTO(){

    }

    public CommoditiesBatchDTO(int rbId, int raavareId, double maengde) {
        this.rbId = rbId;
        this.raavareId = raavareId;
        this.maengde = maengde;
    }

    public int getRbId() {
        return rbId;
    }

    public void setRbId(int rbId) {
        this.rbId = rbId;
    }

    public int getRaavareId() {
        return raavareId;
    }

    public void setRaavareId(int raavareId) {
        this.raavareId = raavareId;
    }

    public double getMaengde() {
        return maengde;
    }

    public void setMaengde(double maengde) {
        this.maengde = maengde;
    }

}
```

```java
package dtu.client.ui;

import com.google.gwt.event.dom.client.ClickEvent;

public class CommoditiesBatchMenu extends Composite {
    private HorizontalPanel hPanel = new HorizontalPanel();


    public CommoditiesBatchMenu(final MainView main, final int level) {
        initWidget(this.hPanel);

        Anchor opret_RaavareBatch = new Anchor("Opret råvarebatch");
        hPanel.add(opret_RaavareBatch);
        opret_RaavareBatch.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.addCommoditiesBatch();
            }
        });

        Anchor vis_RaavareBatch = new Anchor("Vis råvarebatch");
        hPanel.add(vis_RaavareBatch);
        vis_RaavareBatch.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showCommoditiesBatch();
            }
        });

        Anchor back = new Anchor("Tilbage");
        hPanel.add(back);
        back.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.clearContentView();
                if (level == 4){
                    main.showAdministratorMenu(level);
                }
                else if (level == 3){
                    main.showpharmacistMenu(level);
                }

                else if (level == 2){
                    main.showForemanMenu(level);
                }
            }
        });
    }
}


```

```java
package dtu.shared;

import java.io.Serializable;

/**

 * Operatør Data Transfer Object

 */

public class CommoditiesDTO implements Serializable {

    /** Operatør id i området 1-99999999. Vælges af brugerne */
    int raavare_ID;

    /** Operatør navn min. 2 max. 20 karakterer */
    String raavare_Navn;

    /** Operatør initialer min. 2 max. 3 karakterer */
    String leverandoer;

    public CommoditiesDTO() {
    }

    public CommoditiesDTO(int raavare_ID, String raavare_Navn, String leverandoer) {
        this.raavare_ID= raavare_ID;
        this.raavare_Navn = raavare_Navn;
        this.leverandoer = leverandoer;
    }


    public int getRvrId() {
        return raavare_ID;
    }

    public void setRvrId(int raavare_ID) {
        this.raavare_ID = raavare_ID;
    }

    public String getRvrNavn() {
        return raavare_Navn;
    }

    public void setRvrNavn(String raavare_Navn) {
        this.raavare_Navn = raavare_Navn;
    }

    public String getlvr() {
        return leverandoer;
    }

    public void setlvr (String leverandoer) {
        this.leverandoer = leverandoer;
    }


}
```

```java
package dtu.client.ui;

import com.google.gwt.event.dom.client.ClickEvent;

public class CommoditiesMenu extends Composite {
    private HorizontalPanel hPanel = new HorizontalPanel();


    public CommoditiesMenu(final MainView main, final int level) {
        initWidget(this.hPanel);

        Anchor tilfoej_raavare = new Anchor("Tilføj råvare");
        hPanel.add(tilfoej_raavare);
        tilfoej_raavare.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.addCommodities();
            }
        });

        Anchor ret_raavare = new Anchor("Ret/vis råvarer");
        hPanel.add(ret_raavare);
        ret_raavare.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.editCommodities();
            }
        });

        Anchor back = new Anchor("Tilbage");
        hPanel.add(back);
        back.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.clearContentView();
                if(level==4){
                    main.showAdministratorMenu(level);
                }
                else if(level == 3){
                    main.showpharmacistMenu(level);
                }

                else if(level == 2){
                    main.showForemanMenu(level);
                }
            }
    });
    }
}


```

```java
package dtu.client.ui;

import java.util.List;

public class CommodityBatchBrowseView extends Composite {
    DatabaseServiceClientImpl clientImpl;
    VerticalPanel browsePanel;
    FlexTable t;

    public CommodityBatchBrowseView(DatabaseServiceClientImpl clientImpl) {
        this.clientImpl = clientImpl;
        browsePanel = new VerticalPanel();
        initWidget(this.browsePanel);

        HorizontalPanel topPanel = new HorizontalPanel();
        Label pageTitleLbl = new Label("Vis raavarebatches");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.setWidth("450px");
        topPanel.add(pageTitleLbl);
        topPanel.addStyleName("spacing-vertical");
        browsePanel.add(topPanel);

        t = new FlexTable();

        t.getFlexCellFormatter().setWidth(0, 0, "8em");
        t.getFlexCellFormatter().setWidth(0, 1, "10em");
        t.getFlexCellFormatter().setWidth(0, 2, "10em");

        t.addStyleName("FlexTable");
        t.getRowFormatter().addStyleName(0,"FlexTable-Header");

        // set headers in flextable
        t.setText(0, 1, "Raavarebatch ID");
        t.setText(0, 0, "Raavare ID");
        t.setText(0, 2, "Maengde");

        getRaavareBatches();
        browsePanel.add(t);
    }

    private void getRaavareBatches() {
        clientImpl.service.getCommodityBatch(new
  AsyncCallback<List<CommoditiesBatchDTO>>() {

            @Override
            public void onFailure(Throwable caught) {
                Window.alert("Server fejl!" + caught.getMessage());
            }

            @Override
            public void onSuccess(List<CommoditiesBatchDTO> result) {
                for (int i=0; i < result.size(); i++) {
                    t.setText(i+1, 0,""+ result.get(i).getRbId());
                    t.setText(i+1, 1, ""+result.get(i).getRaavareId());
                    t.setText(i+1, 2,""+ result.get(i).getMaengde());
                }
            }
        });
    }
}

```

```java
package dtu.client.ui;

import java.util.List;


public class CommodityEditView extends Composite {
    VerticalPanel editPanel;
    FlexTable t;

    // editing text boxes
    TextBox nameTxt;
    TextBox lvrTxt;

    boolean nameValid = true;
    boolean lvrValid = true;

    int eventRowIndex;

    DatabaseServiceClientImpl clientImpl;

    // operator list
    List<CommoditiesDTO> raavarer;

    // previous cancel anchor
    Anchor previousCancel = null;


    public CommodityEditView(DatabaseServiceClientImpl clientImpl) {
        this.clientImpl = clientImpl;
        editPanel = new VerticalPanel();
        initWidget(this.editPanel);
        HorizontalPanel topPanel = new HorizontalPanel();
        Label pageTitleLbl = new Label("Ret råvarer");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.setWidth("450px");
        topPanel.add(pageTitleLbl);
        topPanel.addStyleName("spacing-vertical");
        editPanel.add(topPanel);

        t = new FlexTable();

        // adjust column widths
        t.getFlexCellFormatter().setWidth(0, 0, "140px");
        t.getFlexCellFormatter().setWidth(0, 1, "150px");
        t.getFlexCellFormatter().setWidth(0, 2, "150px");

        // style table
        t.addStyleName("FlexTable");
        t.getRowFormatter().addStyleName(0,"FlexTable-Header");

        // set headers in flextable
        t.setText(0, 0, "ID");
        t.setText(0, 1, "Navn");
        t.setText(0, 2, "Leverandør");

        getRaavarer();

        editPanel.add(t);

        // text boxes
        nameTxt = new TextBox();
        nameTxt.setWidth("140px");
```

```
83          lvrTxt = new TextBox();
84          lvrTxt.setWidth("140px");
85      }
86
87      private void getRaavarer() {
88          clientImpl.service.getCommodity(new AsyncCallback<List<CommoditiesDTO>>() {
89
90              @Override
91              public void onFailure(Throwable caught) {
92                  Window.alert("Server fejl! " + caught.getMessage());
93              }
94
95              @Override
96              public void onSuccess(List<CommoditiesDTO> result) {
97                  // populate table and add delete anchor to each row
98                  for (int rowIndex=0; rowIndex < result.size(); rowIndex++) {
99                      t.setText(rowIndex+1, 0, ""+result.get(rowIndex).getRvrId());
100                     t.setText(rowIndex+1, 1, result.get(rowIndex).getRvrNavn());
101                     t.setText(rowIndex+1, 2, result.get(rowIndex).getlvr());
102                     Anchor edit = new Anchor("edit");
103                     t.setWidget(rowIndex+1, 3, edit);
104                     edit.addClickHandler(new EditHandler());
105                 }
106             }
107         });
108     }
109
110     private class EditHandler implements ClickHandler {
111         public void onClick(ClickEvent event) {
112
113             // if previous edit open - force cancel operation
114             if (previousCancel != null)
115                 previousCancel.fireEvent(new ClickEvent(){});
116
117             // get rowindex where event happened
118             eventRowIndex = t.getCellForEvent(event).getRowIndex();
119
120             // populate textboxes
121             nameTxt.setText(t.getText(eventRowIndex, 1));
122             lvrTxt.setText(t.getText(eventRowIndex, 2));
123
124             // show text boxes for editing
125             t.setWidget(eventRowIndex, 1, nameTxt);
126             t.setWidget(eventRowIndex, 2, lvrTxt);
127
128             // start editing here
129             nameTxt.setFocus(true);
130
131             // get edit anchor ref for cancel operation
132             final Anchor edit =  (Anchor) event.getSource();
133
134             // get textbox contents for cancel operation
135             final String name = nameTxt.getText();
136             final String ini = lvrTxt.getText();
137
138
139             final Anchor ok = new Anchor("ok");
140             ok.addClickHandler(new ClickHandler() {
141
142                 @Override
143                 public void onClick(ClickEvent event) {
144
```

```
145                     // fill DTO with id and new values
146                     CommoditiesDTO commoditiesDTO = new
    CommoditiesDTO(Integer.parseInt(t.getText(eventRowIndex, 0)), nameTxt.getText(),
    lvrTxt.getText());
147
148
149                     clientImpl.service.updateCommodity(commoditiesDTO, new
    AsyncCallback<Void>() {
150
151                         @Override
152                         public void onSuccess(Void result) {
153                             // remove inputboxes
154                             t.setText(eventRowIndex, 1, nameTxt.getText());
155                             t.setText(eventRowIndex, 2, lvrTxt.getText());
156                         }
157
158                         @Override
159                         public void onFailure(Throwable caught) {
160                             Window.alert("Server fejl!" + caught.getMessage());
161                         }
162
163                     });
164                     // restore edit link
165                     t.setWidget(eventRowIndex, 3, edit);
166                     t.clearCell(eventRowIndex, 4);
167                     previousCancel = null;
168                 }
169             });
170
171             Anchor cancel = new Anchor("cancel");
172             previousCancel = cancel;
173             cancel.addClickHandler(new ClickHandler() {
174
175                 @Override
176                 public void onClick(ClickEvent event) {
177
178                     // restore original content of textboxes and rerun input
    validation
179
180                     nameTxt.setText(name);
181                     nameTxt.fireEvent(new KeyUpEvent() {}); // validation
182
183                     lvrTxt.setText(ini);
184                     lvrTxt.fireEvent(new KeyUpEvent() {});  // validation
185
186                     t.setText(eventRowIndex, 1, name);
187                     t.setText(eventRowIndex, 2, ini);
188                     // restore edit link
189                     t.setWidget(eventRowIndex, 3, edit);
190                     t.clearCell(eventRowIndex, 4);
191
192                     previousCancel = null;
193                 }
194
195             });
196
197
198             nameTxt.addKeyUpHandler(new KeyUpHandler(){
199
200                 @Override
201                 public void onKeyUp(KeyUpEvent event) {
202                     if (!FieldVerifier.isValidName(nameTxt.getText())) {
```

```
203                         nameTxt.setStyleName("gwt-TextBox-invalidEntry");
204                         nameValid = false;
205                     }
206                     else {
207                         nameTxt.removeStyleName("gwt-TextBox-invalidEntry");
208                         nameValid = true;
209                     }
210 
211                     if (nameValid&&lvrValid)
212                         t.setWidget(eventRowIndex, 3, ok);
213                     else
214                         t.setText(eventRowIndex, 3, "ok");
215                 }
216 
217             });
218 
219             lvrTxt.addKeyUpHandler(new KeyUpHandler(){
220 
221                 @Override
222                 public void onKeyUp(KeyUpEvent event) {
223                     if (!FieldVerifier.isValidName(lvrTxt.getText())) {
224                         lvrTxt.setStyleName("gwt-TextBox-invalidEntry");
225                         lvrValid = false;
226                     }
227                     else {
228                         lvrTxt.removeStyleName("gwt-TextBox-invalidEntry");
229                         lvrValid = true;
230                     }
231 
232                     if (nameValid&&lvrValid)
233                         t.setWidget(eventRowIndex, 3, ok);
234                     else
235                         t.setText(eventRowIndex, 3, "ok");
236                 }
237 
238             });
239 
240             // showing ok and cancel widgets
241             t.setWidget(eventRowIndex, 3, ok);
242             t.setWidget(eventRowIndex, 4, cancel);
243         }
244     }
245 }
246 
```

```java
package dtu.client.ui;

import com.google.gwt.user.client.Window;



public class ContentView extends Composite {

    // reference to remote data layer
    private DatabaseServiceClientImpl clientImpl;

    VerticalPanel contentPanel;

    public ContentView() {
    }

    public ContentView(DatabaseServiceClientImpl clientImpl) {
        this.clientImpl = clientImpl;
        contentPanel = new VerticalPanel();
        initWidget(this.contentPanel);
    }

    // Sub views
    public void openWelcomeView(MainView mainView) {
        contentPanel.clear();
        WelcomeView welcomeView = new WelcomeView(clientImpl, mainView);
        contentPanel.add(welcomeView);
    }

    public void clearView() {
        contentPanel.clear();
    }

    // Recipecompontent (receptkomponenter) views
    public void openAddRecipeComponentsView() {
        contentPanel.clear();
        RecipeComponentAddView addView = new RecipeComponentAddView(clientImpl);
        contentPanel.add(addView);
    }

    public void openRecipeComponentsBrowseView() {
        contentPanel.clear();
        RecipeComponentBrowseView browseView = new
RecipeComponentBrowseView(clientImpl);
        contentPanel.add(browseView);
    }

    // Recipe (recepter) views
    public void openAddRecipeView() {
        contentPanel.clear();
        RecipeAddView addView = new RecipeAddView(clientImpl);
        contentPanel.add(addView);
    }

    public void openRecipeBrowseView() {
        contentPanel.clear();
        RecipeBrowseView browseView = new RecipeBrowseView(clientImpl);
        contentPanel.add(browseView);
    }

    // User views
    public void openUserAddView() {
```

```
67          contentPanel.clear();
68          UserAddView userAddView = new UserAddView(clientImpl);
69          contentPanel.add(userAddView);
70      }
71
72      public void openBrowseUsersView() {
73          contentPanel.clear();
74          UserBrowseView userBrowseView = new UserBrowseView(clientImpl);
75          contentPanel.add(userBrowseView);
76      }
77
78      public void openDeleteUsersView() {
79          contentPanel.clear();
80          UserDeleteView userDeleteView = new UserDeleteView(clientImpl);
81          contentPanel.add(userDeleteView);
82      }
83
84      public void openEditUsersView() {
85          contentPanel.clear();
86          UserEditView userEditView = new UserEditView(clientImpl);
87          contentPanel.add(userEditView);
88      }
89
90      // Commidities (råvarer) views
91      public void openEditCommoditiesView() {
92          contentPanel.clear();
93          CommodityEditView editView = new CommodityEditView(clientImpl);
94          contentPanel.add(editView);
95      }
96
97      public void openAddCommoditiesView() {
98          contentPanel.clear();
99          CommoditiesAddView addView = new CommoditiesAddView(clientImpl);
100         contentPanel.add(addView);
101     }
102
103     // Commodities batch (råvarebatches) views
104     public void openCommoditiesBatchBrowseView() {
105         contentPanel.clear();
106         CommodityBatchBrowseView browseView = new
    CommodityBatchBrowseView(clientImpl);
107         contentPanel.add(browseView);
108     }
109
110     public void openAddCommoditiesBatchView() {
111         contentPanel.clear();
112         CommoditiesBatchAddView addView = new CommoditiesBatchAddView(clientImpl);
113         contentPanel.add(addView);
114     }
115
116     // Product batch views
117     public void openProductBatchBrowseView() {
118         contentPanel.clear();
119         ProductBatchBrowseView browseView = new ProductBatchBrowseView(clientImpl);
120         contentPanel.add(browseView);
121     }
122
123     public void openAddProductBatchView() {
124         contentPanel.clear();
125         ProductBatchAddView addView = new ProductBatchAddView(clientImpl);
126         contentPanel.add(addView);
127     }
```

```
128 }
129
```

```java
package dtu.shared;

/**
 * Data Access Layer Exception is thrown when errors occur in the data layer
 * Exception is serializable and hence DALException is also serializable
 */
public class DALException extends Exception {
    private static final long serialVersionUID = 1L;

    // must have a desfult constructor
    public DALException() {

    }

    public DALException(String s) {
        super(s);
    }
}
```

```java
package dtu.server.dal;

import java.sql.Connection;

public class DAO extends RemoteServiceServlet implements KartotekService {

    private static final String URL = "jdbc:mysql://62.79.16.16/grp16";
//  private static final String URL = "jdbc:mysql://localhost/cdio_3";
    private static final String USERNAME = "grp16";
    private static final String PASSWORD = "ZHnPq74Y";

    private Connection connection = null; // manages connection

    private PreparedStatement loginStmt = null;

    //user
    private PreparedStatement createUserStmt = null;
    private PreparedStatement updateUserStmt = null;
    private PreparedStatement getUserStmt = null;
    private PreparedStatement getSizeStmt = null;
    private PreparedStatement deleteUserStmt = null;

    //R?vareBatch
    private PreparedStatement getCommoditiesBatchStmt = null;
    private PreparedStatement createCommoditiesBatchStmt = null;

    //R?vare
    private PreparedStatement createCommoditiesStmt = null;
    private PreparedStatement updateCommoditiesStmt = null;
    private PreparedStatement getCommoditiesStmt = null;

    //Recept
    private PreparedStatement getRecipeStmt = null;
    private PreparedStatement createRecipeStmt = null;

    // Receptkomponent
    private PreparedStatement getRecipeComponentStmt = null;
    private PreparedStatement createRecipeComponentStmt = null;

    //ProduktBatch
    private PreparedStatement getProductBatchStmt = null;
    private PreparedStatement createProductBatchStmt = null;



    public DAO() throws DALException {
        try
        {
            connection =
                    DriverManager.getConnection( URL, USERNAME, PASSWORD );

            // create query that add an operator to kartotek
            createUserStmt =
                    connection.prepareStatement( "INSERT INTO brugere " +
                            "(opr_navn, ini, cpr, password, active, level) " +
                            "VALUES ( ?, ?, ?, ?, ?, ?)" );

            // create query that updates an operator
            updateUserStmt = connection.prepareStatement(
                    "UPDATE brugere SET opr_navn = ?, ini = ?, cpr = ?, password =
  ?, active = ?, level = ? WHERE opr_id = ?" );

```

```
79              // create query that get all operators in kartotek
80              getUserStmt = connection.prepareStatement(
81                      "SELECT * FROM brugere");
82
83              // create query that gets size of kartotek
84              getSizeStmt = connection.prepareStatement(
85                      "SELECT COUNT(*) FROM brugere");
86
87              // create query that deletes a operator in kartotek
88 //           deleteOperatorStmt = connection.prepareStatement(
89 //                   "DELETE FROM brugere WHERE id =  ?");
90              deleteUserStmt = connection.prepareStatement(
91                      "UPDATE brugere SET active = 0 WHERE opr_id = ?");
92
93              loginStmt = connection.prepareStatement(
94                      "SELECT * FROM brugere WHERE opr_id = ? AND password = ?");
95
96
97              // R?VARER!!!
98              // create query that add an raavare to kartotek
99              createCommoditiesStmt =
100                     connection.prepareStatement( "INSERT INTO raavare " +
101                             "(raavare_id, raavare_navn, leverandoer) " +
102                             "VALUES ( ?, ?, ?)" );
103
104             // create query that updates an raavare
105             updateCommoditiesStmt = connection.prepareStatement(
106                     "UPDATE raavare SET Raavare_navn = ?, leverandoer = ? WHERE
    raavare_id = ?" );
107
108             // create query that get all raavarer in kartotek
109             getCommoditiesStmt = connection.prepareStatement(
110                     "SELECT * FROM raavare");
111
112             createRecipeStmt = connection.prepareStatement( "INSERT INTO recept " +
113                     "(recept_Id, recept_navn) " +
114                     "VALUES (?, ?)" );
115
116             getRecipeStmt = connection.prepareStatement(
117                     "SELECT * FROM recept");
118
119             // RECEPTKOMPONENT
120
121             //Create receptkomponent Query to make table
122             createRecipeComponentStmt = connection.prepareStatement( "INSERT INTO
    receptkomponent " +
123                     "(recept_Id, raavare_id, nom_Netto, tolerance) " +
124                     "VALUES (?, ?, ?, ?)" );
125             //Get receptkomponent Query
126             getRecipeComponentStmt = connection.prepareStatement(
127                     "SELECT * FROM receptkomponent");
128
129             //R?VAREBATCHES
130
131             //Create R?vareBatch query
132             createCommoditiesBatchStmt =
133                     connection.prepareStatement( "INSERT INTO Raavarebatch " +
134                             "(Raavare_ID, Rb_Id, maengde) " +
135                             "VALUES ( ?, ?, ?)" );
136
137             getCommoditiesBatchStmt = connection.prepareStatement(
138                     "SELECT * FROM Raavarebatch");
```

```
139
140             getProductBatchStmt = connection.prepareStatement(
141                     "SELECT * FROM ProduktBatch");
142
143             createProductBatchStmt =
144                     connection.prepareStatement( "INSERT INTO ProduktBatch " +
145                             "(pb_id, status, recept_id) " +
146                             "VALUES ( ?, ?, ?)" );
147
148         }
149         catch ( SQLException sqlException )
150         {
151             throw new DALException("Kan ikke oprette forbindelse til database! " +
    sqlException.getMessage());
152         }
153     }
154
155
156     @Override
157     public void deleteUser(int id) throws DALException {
158         try {
159             deleteUserStmt.setInt(1, id);
160             deleteUserStmt.executeUpdate();
161         } catch (SQLException e) {
162             throw new DALException(" \"deleteOperator\" fejlede" + e.getMessage());
163         }
164     }
165
166     @Override
167     public List<UserDTO> getUser() throws DALException {
168         List<UserDTO> results = null;
169         ResultSet resultSet = null;
170
171         try
172         {
173             resultSet = getUserStmt.executeQuery();
174             results = new ArrayList< UserDTO >();
175
176             while (resultSet.next())
177             {
178                 results.add( new UserDTO(
179                         resultSet.getInt("opr_id"),
180                         resultSet.getString("opr_navn"),
181                         resultSet.getString("ini"),
182                         resultSet.getString("cpr"),
183                         resultSet.getString("password"),
184                         resultSet.getInt("active"),
185                         resultSet.getInt("level")));
186             }
187         }
188         catch ( SQLException e )
189         {
190             throw new DALException(" \"getOperators\" fejlede " + e.getMessage());
191         }
192         finally
193         {
194             try
195             {
196                 resultSet.close();
197             }
198             catch ( SQLException sqlException )
199             {
```

```
200                 sqlException.printStackTrace();
201                 close();
202             }
203         }
204         return results;
205     }
206 
207     @Override
208     public int getSize() throws DALException {
209         try {
210             ResultSet rs = null;
211             rs = getSizeStmt.executeQuery();
212             rs.next();
213             return rs.getInt(1);
214         } catch (SQLException e) {
215             throw new DALException(" \"getSize\" fejlede " + e.getMessage());
216         }
217     }
218 
219     @Override
220     public void createUser(UserDTO p) throws Exception {
221         try {
222             createUserStmt.setString(1, p.getOprNavn());
223             createUserStmt.setString(2, p.getIni());
224             createUserStmt.setString(3, p.getCpr());
225             createUserStmt.setString(4, p.getPassword());
226             createUserStmt.setString(5, p.getActive());
227             createUserStmt.setString(6, p.getLevel());
228             createUserStmt.executeUpdate();
229         } catch (SQLException e) {
230             throw new DALException(" \"createOperator\" fejlede " +
    e.getMessage());
231         }
232     }
233 
234     @Override
235     public void updateUser(UserDTO p) throws Exception {
236         try {
237             updateUserStmt.setString(1, p.getOprNavn());
238             updateUserStmt.setString(2, p.getIni());
239             updateUserStmt.setString(3, p.getCpr());
240             updateUserStmt.setString(4, p.getPassword());
241             updateUserStmt.setString(5, p.getActive());
242             updateUserStmt.setString(6, p.getLevel());
243             updateUserStmt.setInt(7, p.getOprId());
244             updateUserStmt.executeUpdate();
245         } catch (SQLException e) {
246             throw new DALException(" \"updateOperator\" fejlede: " +
    e.getMessage());
247         }
248     }
249 
250     public void close() {
251         try {
252             connection.close();
253         } // end try
254         catch (SQLException sqlException) {
255             sqlException.printStackTrace();
256         }
257     }
258 
259 
```

```
260     @Override
261     public int login(int id, String password) throws DALException{
262         ResultSet resultSet = null;
263         try
264         {
265 //          loginStmt.setInt(1, 1);
266 //          loginStmt.setString(2, "HelloKitty");
267             loginStmt.setInt(1, id);
268             loginStmt.setString(2, password);
269             try {
270                 resultSet = loginStmt.executeQuery();
271             } catch (Exception e) {
272                 throw new DALException(" \"Login\" fejlede " + e.getMessage());
273             }
274 
275 
276             if (resultSet.first()) return resultSet.getInt(7);
277 
278         }
279         catch ( SQLException e )
280         {
281             throw new DALException(" \"Login\" fejlede " + e.getMessage());
282         }
283         finally
284         {
285             try
286             {
287                 resultSet.close();
288             }
289             catch ( SQLException sqlException )
290             {
291                 sqlException.printStackTrace();
292                 close();
293             }
294         }
295         return 0;
296     }
297 
298 
299     @Override
300     public void createCommodity(CommoditiesDTO p) throws Exception {
301         try {
302             createCommoditiesStmt.setInt(1, p.getRvrId());
303             createCommoditiesStmt.setString(2, p.getRvrNavn());
304             createCommoditiesStmt.setString(3, p.getlvr());
305 
306             createCommoditiesStmt.executeUpdate();
307         } catch (SQLException e) {
308             throw new DALException(" \"createRaavare\" fejlede " + e.getMessage());
309         }
310     }
311 
312 
313     @Override
314     public void updateCommodity(CommoditiesDTO p) throws Exception {
315         try {
316             updateCommoditiesStmt.setString(1, p.getRvrNavn());
317             updateCommoditiesStmt.setString(2, p.getlvr());
318             updateCommoditiesStmt.setInt(3, p.getRvrId());
319             updateCommoditiesStmt.executeUpdate();
320         } catch (SQLException e) {
321             throw new DALException(" \"updateRaavare\" fejlede: " +
```

```
   e.getMessage());
322         }
323     }
324
325
326     @Override
327     public List<CommoditiesDTO> getCommodity() throws Exception {
328         List<CommoditiesDTO> results = null;
329         ResultSet resultSet = null;
330
331         try
332         {
333             resultSet = getCommoditiesStmt.executeQuery();
334             results = new ArrayList< CommoditiesDTO >();
335
336             while (resultSet.next())
337             {
338                 results.add( new CommoditiesDTO(
339                         resultSet.getInt("raavare_ID"),
340                         resultSet.getString("raavare_navn"),
341                         resultSet.getString("leverandoer")));
342             }
343         }
344         catch ( SQLException e )
345         {
346             throw new DALException(" \"getRaavarer\" fejlede " + e.getMessage());
347         }
348         finally
349         {
350             try
351             {
352                 resultSet.close();
353             }
354             catch ( SQLException sqlException )
355             {
356                 sqlException.printStackTrace();
357                 close();
358             }
359         }
360         return results;
361     }
362
363     //CREATE RECEPT
364
365     @Override
366     public void createRecipeComponent(ReceptKomponentDTO p) throws Exception {
367         try {
368             createRecipeComponentStmt.setInt(1, p.getRcpId());
369             createRecipeComponentStmt.setInt(2, p.getRvrId());
370             createRecipeComponentStmt.setDouble(3, p.getNomNetto());
371             createRecipeComponentStmt.setDouble(4, p.getTolerance());
372             createRecipeComponentStmt.executeUpdate();
373         } catch (SQLException e) {
374             throw new DALException(" \"createReceptKomponent\" fejlede "+
   e.getMessage());
375         }
376     }
377
378     // Vis RECEPT
379
380     @Override
381     public List<ReceptKomponentDTO> getRecipeComponent() throws DALException {
```

```
382         List<ReceptKomponentDTO> results = null;
383         ResultSet resultSet = null;
384         try
385         {
386             resultSet = getRecipeComponentStmt.executeQuery();
387             results = new ArrayList< ReceptKomponentDTO >();
388 
389             while (resultSet.next())
390             {
391                 results.add( new ReceptKomponentDTO(
392                         resultSet.getInt("Recept_id"),
393                         resultSet.getInt("Raavare_id"),
394                         resultSet.getDouble("Nom_Netto"),
395                         resultSet.getDouble("Tolerance")
396                         ));
397             }
398         }
399         catch ( SQLException e )
400         {
401             throw new DALException(" \"getRecepts\" fejlede " + e.getMessage());
402         }
403         finally
404         {
405             try
406             {
407                 resultSet.close();
408             }
409             catch ( SQLException sqlException )
410             {
411                 sqlException.printStackTrace();
412                 close();
413             }
414         }
415         return results;
416     }
417 
418     //R?VAREBATCH
419     //Create tuple
420     @Override
421     public void createCommodityBatch(CommoditiesBatchDTO p) throws Exception {
422         try {
423             createCommoditiesBatchStmt.setInt(1, p.getRaavareId());
424             createCommoditiesBatchStmt.setInt(2, p.getRbId());
425             createCommoditiesBatchStmt.setDouble(3, p.getMaengde());
426             createCommoditiesBatchStmt.executeUpdate();
427         } catch (SQLException e) {
428             throw new DALException(" \"createRaavarebatch\" fejlede " +
    e.getMessage());
429         }
430     }
431 
432     @Override
433     public List<CommoditiesBatchDTO> getCommodityBatch() throws DALException {
434         List<CommoditiesBatchDTO> results = null;
435         ResultSet resultSet = null;
436         try
437         {
438             resultSet = getCommoditiesBatchStmt.executeQuery();
439             results = new ArrayList< CommoditiesBatchDTO >();
440 
441             while (resultSet.next())
442             {
```

```
443                 results.add(new CommoditiesBatchDTO(
444                         resultSet.getInt("rb_id"),
445                         resultSet.getInt("Raavare_id"),
446                         resultSet.getDouble("maengde")
447                         ));
448             }
449         }
450         catch ( SQLException e )
451         {
452             throw new DALException(" \"getRaavareBatch\" fejlede " +
    e.getMessage());
453         }
454         finally
455         {
456             try
457             {
458                 resultSet.close();
459             }
460             catch ( SQLException sqlException )
461             {
462                 sqlException.printStackTrace();
463                 close();
464             }
465         }
466         return results;
467     }
468 
469     @Override
470     public void createRecipe(RecipeDTO p) throws Exception {
471         try {
472             createRecipeStmt.setInt(1, p.getRcpId());
473             createRecipeStmt.setString(2, p.getRecept_Navn());
474             createRecipeStmt.executeUpdate();
475         } catch (SQLException e) {
476             throw new DALException(" \"createRecept\" fejlede "+ e.getMessage());
477         }
478     }
479 
480 
481     @Override
482     public List<RecipeDTO> getRecipe() throws DALException {
483         List<RecipeDTO> results = null;
484         ResultSet resultSet = null;
485         try
486         {
487             resultSet = getRecipeStmt.executeQuery();
488             results = new ArrayList< RecipeDTO >();
489 
490             while (resultSet.next())
491             {
492                 results.add( new RecipeDTO(
493                         resultSet.getInt("Recept_id"),
494                         resultSet.getString("Recept_navn")
495                         ));
496             }
497         }
498         catch ( SQLException e )
499         {
500             throw new DALException(" \"getRecepts\" fejlede " + e.getMessage());
501         }
502         finally
503         {
```

```
504             try
505             {
506                 resultSet.close();
507             }
508             catch ( SQLException sqlException )
509             {
510                 sqlException.printStackTrace();
511                 close();
512             }
513         }
514         return results;
515     }
516 
517 
518     @Override
519     public void createProductBatch(ProductBatchDTO p) throws Exception {
520         try {
521             createProductBatchStmt.setInt(1, p.getPb_ID());
522             createProductBatchStmt.setInt(2, p.getStatus());
523             createProductBatchStmt.setInt(3, p.getRecept_id());
524             createProductBatchStmt.executeUpdate();
525         } catch (SQLException e) {
526             throw new DALException(" \"createProduktBatcht\" fejlede "+
    e.getMessage());
527         }
528     }
529 
530     @Override
531     public List<ProductBatchDTO> getProductBatch() throws DALException {
532         List<ProductBatchDTO> results = null;
533         ResultSet resultSet = null;
534         try
535         {
536             resultSet = getProductBatchStmt.executeQuery();
537             results = new ArrayList< ProductBatchDTO >();
538 
539             while (resultSet.next())
540             {
541                 results.add( new ProductBatchDTO(
542                         resultSet.getInt("Pb_Id"),
543                         resultSet.getInt("Status"),
544                         resultSet.getInt("Recept_Id")
545                         ));
546             }
547         }
548         catch ( SQLException e )
549         {
550             throw new DALException(" \"getProduktBatch\" fejlede " +
    e.getMessage());
551         }
552         finally
553         {
554             try
555             {
556                 resultSet.close();
557             }
558             catch ( SQLException sqlException )
559             {
560                 sqlException.printStackTrace();
561                 close();
562             }
563         }
```

```
564         return results;
565     }
566 }
567
```

```java
package tests;

import static org.junit.Assert.*;

public class DAOTest {
    static String ListeB = "";
    static String ListeA = "";

    @Test
    public void test() throws Exception {
DAO dao = new DAO();

//      System.out.println("Vi definere en liste med alle superbrugere");
        List<UserDTO> ListBefore = dao.getUser();
        for (int i = 0; i < ListBefore.size(); i++) {
            ListeB = ListeB + ListBefore.get(i).getOprNavn();
            ListeB += ", ";
        }

        UserDTO op = new UserDTO("Jens Peter", "JP", "0987654444", "PassJP10", 1,
 4);
        dao.createUser(op);

        List<UserDTO> ListAfter = dao.getUser();
        for (int i = 0; i < ListAfter.size(); i++) {
            ListeA = ListeA + ListAfter.get(i).getOprNavn();
            ListeA += ", ";
        }

        if(ListeA.equals(ListeB)){
            fail();
        }
        dao.deleteUser(op.getOprId());
        if(!dao.getUser().get(0).getActive().equals(op.getActive())){
            fail();
        };

        String name = "Jens Christian";
        op.setOprNavn(name);
        dao.updateUser(op);
        if(!op.getOprNavn().equals(name)){
            fail();
        }
    }

}
```

```java
package dtu.shared;

/**
 * <p>
 * FieldVerifier validates that the name the user enters is valid.
 * </p>
 * <p>
 * This class is in the <code>shared</code> package because we use it in both
 * the client code and on the server. On the client, we verify that the name is
 * valid before sending an RPC request so the user doesn't have to wait for a
 * network round trip to get feedback. On the server, we verify that the name is
 * correct to ensure that the input is correct regardless of where the RPC
 * originates.
 * </p>
 * <p>
 * When creating a class that is used on both the client and the server, be sure
 * that all code is translatable and does not use native JavaScript. Code that
 * is not translatable (such as code that interacts with a database or the file
 * system) cannot be compiled into client-side JavaScript. Code that uses native
 * JavaScript (such as Widgets) cannot be run on the server.
 * </p>
 */
public class FieldVerifier {

    /**
     * Verifies that the specified name is valid for our service.
     *
     * In this example, we only require that the name is at least four
     * characters. In your application, you can use more complex checks to ensure
     * that usernames, passwords, email addresses, URLs, and other fields have the
     * proper syntax.
     *
     * @param name the name to validate
     * @return true if valid, false if invalid
     */

    public static boolean isValidID(String id) {
        if (id == null) return false;
        if (id.length() == 0) return false;
        try {
            if (Integer.valueOf(id) >= 1 && Integer.valueOf(id) <= 99999999) return
 true;
        } catch (NumberFormatException e) {
            return false;
        }
        return false;
    }

    public static boolean isValidName(String name) {
        if (name == null) {
            return false;
        }
        if (name.length() == 0)
            return false;
        // min. 2 og max. 20 karakterer
        return (name.length() >= 2 && name.length() <= 20);
    }

    public static boolean isValidInitials(String ini) {
        if (ini == null) return false;
        if (ini.length() == 0) return false;
        // min. 2 og max. 3 characters
```

```
62          return (ini.length() == 2 || ini.length() == 3);
63      }
64
65      public static boolean isValidCpr(String cpr) {
66          if (cpr == null) return false;
67          if (cpr.length() == 0) return false;
68          // must be 11 chars (CDIO oplæg siger 10? Ingen bindestreg?)
69          return (cpr.length() == 10);
70      }
71
72  //  public static boolean isValidPass(String pass) {
73  //      if (pass == null) return false;
74  //      if (pass.length() == 0) return false;
75  //      // min. 7 & max. 8 characters
76  //      return (pass.length() == 7 || pass.length() == 8);
77  //  }
78
79      // Password validation according to DTU's password rules (Taken from our CDIO1
    (Copyright CDIO group 16, CDIO1, 2015))
80      public static boolean isValidPass(String input) {
81          if (input == null) return false;
82          if (input.length() == 0) return false;
83
84          char[] pw = input.toCharArray();
85          int upper = 0, lower = 0, numbers = 0, symbols = 0, categories = 0;
86          boolean valid = true;
87
88          if (input.length() == 7 || input.length() == 8) {
89              for (char chr : pw) {
90                  if (chr >= 65 && chr <= 90) upper++;// A-Z
91                  else if (chr >= 97 && chr <= 122) lower++;// a-z
92                  else if (chr >= 48 && chr <= 57) numbers++;// 0-9
93                  else if (chr == 95 || chr == 46 || chr == 45 || chr == 44 || chr ==
    33 || chr == 63 || chr == 61 || chr == 43) symbols++; // .-,_+!?=
94                  else valid = false;
95              }
96
97              if (upper > 0) categories++;
98              if (lower > 0) categories++;
99              if (numbers > 0) categories++;
100             if (symbols > 0) categories++;
101         }
102         if (categories >= 3 && valid) { // MINDST 3 AF KATEGORIERNE SKAL VÆRE
    REPRÆSENTERET
103             return true; // PASSWORD ER GYLDIGT
104         } else { return false; } // PASSWORD ER UGYLDIGT
105     }
106
107     public static boolean isValidActive(String active) {
108         if (active == null) return false;
109         if (active.length() != 1) return false;
110         // 1 = active, 0 = inactive
111         return (Integer.valueOf(active) == 1 || Integer.valueOf(active) == 0);
112     }
113
114     public static boolean isValidLevel(String level) {
115         if (level == null) return false;
116         if (level.length() != 1) return false;
117         // 1 = operatør, 2 = superbruger
118         return (Integer.valueOf(level) >= 1 && Integer.valueOf(level)<= 4);
119     }
120
```

```
121     public static boolean isValidTolerance(String Decimal){
122         if(Double.valueOf(Decimal)>10) return false;
123         if(Double.valueOf(Decimal)<0.1) return false;
124         return true;
125     }
126 
127     public static boolean isValidNomNetto(String Netto){
128         if(Double.valueOf(Netto)>20) return false;
129         if(Double.valueOf(Netto)<0.05) return false;
130         return true;
131     }
132 
133     public static boolean IsValidMaengde(String netto){
134         if(!Double.valueOf(netto).isNaN()) return true;
135         return false;
136     }
137 
138     public static boolean isValidStatus(String status){
139         if(Integer.valueOf(status)>=0 && Integer.valueOf(status)<=2) return true;
140         return false;
141     }
142 
143 }
144 
```

```java
package dtu.client.ui;

import com.google.gwt.event.dom.client.ClickEvent;

public class ForemanMenu extends Composite {
    private HorizontalPanel hPanel = new HorizontalPanel();

    // receive reference to MainView for call back
    public ForemanMenu(final MainView main, final int level) {
        initWidget(this.hPanel);

        Anchor raavarebatch_administration = new Anchor("Råvarebatch");
        hPanel.add(raavarebatch_administration);
        raavarebatch_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showCommoditiesBatchMenu(level);
            }
        });

        Anchor produktbatch_administration = new Anchor("Produktbatch");
        hPanel.add(produktbatch_administration);
        produktbatch_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showProductBatchMenu(level);           }
        });
    }
}

```

DatabaseServiceClientImpl.java

```java
package dtu.client.service;

import com.google.gwt.core.shared.GWT;
public class DatabaseServiceClientImpl {

    // global reference to service endpoint
    public KartotekServiceAsync service;

    public DatabaseServiceClientImpl(String url) {
        // Set RPC service end point
        this.service = GWT.create(KartotekService.class);
        ServiceDefTarget endpoint = (ServiceDefTarget) this.service;
        endpoint.setServiceEntryPoint(url);
    }
}

```

```java
package dtu.client;

import com.google.gwt.core.client.EntryPoint;

/**
 * Entry point classes define <code>onModuleLoad()</code>.
 */
public class Kartotek implements EntryPoint {

    public void onModuleLoad() {
        new MainView().run();
    }
}

```

```java
package dtu.client.service;

import java.util.List;


@RemoteServiceRelativePath("kartotekservice")
public interface KartotekService extends RemoteService {

    // note: announcing exception makes it possible to communicate
    // user defined exceptions from the server side to the client side
    // otherwise only generic server exceptions will be send back
    // in the onFailure call back method

    // Operat�rer
    void createUser(UserDTO op) throws Exception;
    void deleteUser(int id) throws Exception;
    void updateUser(UserDTO op) throws Exception;
    List<UserDTO> getUser() throws Exception;
    int getSize() throws Exception;

    int login(int id, String password) throws Exception;

    // R�varer
    void createCommodity(CommoditiesDTO op) throws Exception;
    void updateCommodity(CommoditiesDTO op) throws Exception;
    List<CommoditiesDTO> getCommodity() throws Exception;

    //Receptkomponent
    void createRecipeComponent(ReceptKomponentDTO p) throws Exception;
    List<ReceptKomponentDTO> getRecipeComponent() throws DALException;

    //Recept
    void createRecipe(RecipeDTO p) throws Exception;
    List<RecipeDTO> getRecipe() throws DALException;

    //Raavarebatch
    void createCommodityBatch(CommoditiesBatchDTO p) throws Exception;
    List<CommoditiesBatchDTO> getCommodityBatch() throws DALException;

    //ProduktBatch
    void createProductBatch(ProductBatchDTO p) throws Exception;
    List<ProductBatchDTO> getProductBatch() throws DALException;
}

```

```java
package dtu.client.service;

import java.util.List;

public interface KartotekServiceAsync {

    void login(int id, String password, AsyncCallback<Integer> callback);

    // Brugere
    void createUser(UserDTO p, AsyncCallback<Void> callback);

    void updateUser(UserDTO p, AsyncCallback<Void> callback);

    void getUser(AsyncCallback<List<UserDTO>> callback);

    void deleteUser(int id, AsyncCallback<Void> callback);

    void getSize(AsyncCallback<Integer> callback);

    // R?varer
    void createCommodity(CommoditiesDTO p, AsyncCallback<Void> callback);

    void updateCommodity(CommoditiesDTO p, AsyncCallback<Void> callback);

    void getCommodity(AsyncCallback<List<CommoditiesDTO>> callback);

    // Receptkomponent
    void createRecipeComponent(ReceptKomponentDTO p, AsyncCallback<Void> callback);

    void getRecipeComponent(AsyncCallback<List<ReceptKomponentDTO>> callback);

    //Recept
    void createRecipe(RecipeDTO newRecept, AsyncCallback<Void> asyncCallback);

    void getRecipe(AsyncCallback<List<RecipeDTO>> callback);

    //Raavarebatch
    void createCommodityBatch(CommoditiesBatchDTO p, AsyncCallback<Void> callback) ;

    void getCommodityBatch(AsyncCallback<List<CommoditiesBatchDTO>> callback);

    //ProduktBatch
    void createProductBatch(ProductBatchDTO p, AsyncCallback<Void> callback) ;

    void getProductBatch(AsyncCallback<List<ProductBatchDTO>> callback);

}

```

```java
package dtu.client.controller;

import com.google.gwt.core.client.GWT;


public class MainView  {

    // reference to ContentView
    private ContentView contentView;

    // V.2
    // reference to remote data layer
    private DatabaseServiceClientImpl clientImpl;


    public MainView() {

        // V.2
        // add server side implementation of data layer
        clientImpl = new DatabaseServiceClientImpl(GWT.getModuleBaseURL() + "kartotekservice");

        // wrap contentView
        contentView = new ContentView(clientImpl);
        RootPanel.get("section").add(contentView);
    }

    public void run() {
        // show welcome panel
        contentView.openWelcomeView(this);
    }

    public void clearContentView() {
        contentView.clearView();
    }

    public void showUserMenu(int level) {
        // wrap menuView
        RootPanel.get("nav").clear();
        UserAdminMenu m = new UserAdminMenu(this, level);
        RootPanel.get("nav").add(m);
    }

    public void showForemanMenu(int level) {
        // wrap menuView
        RootPanel.get("nav").clear();
        ForemanMenu m = new ForemanMenu(this, level);
        RootPanel.get("nav").add(m);
    }

    public void showpharmacistMenu(int level) {
        // wrap menuView
        RootPanel.get("nav").clear();
        PharmacistMenu m = new PharmacistMenu(this, level);
        RootPanel.get("nav").add(m);
    }

    public void showAdministratorMenu(int level) {
        // wrap menuView
        RootPanel.get("nav").clear();
        AdministratorMenu m = new AdministratorMenu(this, level);
        RootPanel.get("nav").add(m);
```

```
    }

    public void showCommoditiesMenu(int level){
        RootPanel.get("nav").clear();
        CommoditiesMenu m = new CommoditiesMenu(this, level);
        RootPanel.get("nav").add(m);
    }

    public void showRecipeKomponentMenu(int level){
        RootPanel.get("nav").clear();
        RecipeComponentMenu m = new RecipeComponentMenu(this, level);
        RootPanel.get("nav").add(m);
    }

    public void showCommoditiesBatchMenu(int level){
        RootPanel.get("nav").clear();
        CommoditiesBatchMenu m = new CommoditiesBatchMenu(this, level);
        RootPanel.get("nav").add(m);
    }

    public void showProductBatchMenu(int level){
        RootPanel.get("nav").clear();
        ProductBatchMenu m = new ProductBatchMenu(this, level);
        RootPanel.get("nav").add(m);
    }

    // Call back handlers
    public void addUser() {
        contentView.openUserAddView();
    }

    public void addCommodities() {
        contentView.openAddCommoditiesView();
    }

// Recept
    public void addRecipeComponent() {
        contentView.openAddRecipeComponentsView();
    }

    public void addRecipe() {
        contentView.openAddRecipeView();
    }

    public void addCommoditiesBatch() {
        contentView.openAddCommoditiesBatchView();
    }

    public void addProductBatch() {
        contentView.openAddProductBatchView();
    }

    public void showUsers() {
        contentView.openBrowseUsersView();
    }

    public void editUsers() {
        contentView.openEditUsersView();
    }

    public void editCommodities() {
        contentView.openEditCommoditiesView();
```

```
137     }
138
139     public void deleteUsers() {
140         contentView.openDeleteUsersView();
141     }
142
143     public void showRecipes() {
144         contentView.openRecipeBrowseView();
145     }
146
147     public void showCommoditiesBatch() {
148         contentView.openCommoditiesBatchBrowseView();
149     }
150
151     public void showProductBatch() {
152         contentView.openProductBatchBrowseView();
153     }
154
155     public void showRecipeComponents() {
156         contentView.openRecipeComponentsBrowseView();
157     }
158
159     public void showRecipeMenu(int level) {
160         RootPanel.get("nav").clear();
161         RecipeMenu m = new RecipeMenu(this, level);
162         RootPanel.get("nav").add(m);
163     }
164
165
166
167
168
169
170
171 }
172
```

```java
package dtu.client.ui;

import com.google.gwt.event.dom.client.ClickEvent;

public class RecipeMenu extends Composite {
    private HorizontalPanel hPanel = new HorizontalPanel();


    public RecipeMenu(final MainView main, final int level) {
        initWidget(this.hPanel);

        Anchor vis_recept = new Anchor("Vis recepter");
        hPanel.add(vis_recept);
        vis_recept.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showRecipes();
            }
        });

        Anchor tilfoej_recept = new Anchor("Tilføj recept");
        hPanel.add(tilfoej_recept);
        tilfoej_recept.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.addRecipe();
            }
        });


        Anchor back = new Anchor("Tilbage");
        hPanel.add(back);
        back.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.clearContentView();
                if (level == 4){
                    main.showAdministratorMenu(level);
                }
                else if (level == 3){
                    main.showpharmacistMenu(level);
                }

                else if (level == 2){
                    main.showForemanMenu(level);
                }
            }
        });
    }
}


```

```java
package dtu.client.ui;

import java.util.List;

public class UserBrowseView extends Composite {
    DatabaseServiceClientImpl clientImpl;
    VerticalPanel browsePanel;
    static FlexTable t;
    Button showInactiveOps;
    boolean showInactive = false;
    int rowCount;

    public UserBrowseView(DatabaseServiceClientImpl clientImpl) {
        this.clientImpl = clientImpl;
        browsePanel = new VerticalPanel();
        initWidget(this.browsePanel);

        HorizontalPanel topPanel = new HorizontalPanel();
        showInactiveOps = new Button("Vis inaktive brugere");
        Label pageTitleLbl = new Label("Vis brugere");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.setWidth("450px");
        topPanel.add(pageTitleLbl);
        topPanel.add(showInactiveOps);
        topPanel.addStyleName("spacing-vertical");
        browsePanel.add(topPanel);

        t = new FlexTable();

        t.getFlexCellFormatter().setWidth(0, 0, "4em");
        t.getFlexCellFormatter().setWidth(0, 1, "20em");
        t.getFlexCellFormatter().setWidth(0, 2, "3em");
        t.getFlexCellFormatter().setWidth(0, 3, "10em");
        t.getFlexCellFormatter().setWidth(0, 4, "8em");
        t.getFlexCellFormatter().setWidth(0, 5, "3em");
        t.getFlexCellFormatter().setWidth(0, 6, "3em");
        t.addStyleName("FlexTable");
        t.getRowFormatter().addStyleName(0,"FlexTable-Header");

        // set headers in flextable
        t.setText(0, 0, "ID");
        t.setText(0, 1, "Navn");
        t.setText(0, 2, "Initialer");
        t.setText(0, 3, "CPR");
        t.setText(0, 4, "Password");
        t.setText(0, 5, "Aktiv");
        t.setText(0, 6, "Niveau");

        browsePanel.add(t);

        getUsers();
        showInactiveOps.addClickHandler(new ClickHandler() {

            @Override
            public void onClick(ClickEvent event) {
                rowCount = t.getRowCount();
                for (int i = rowCount-1; i > 0; i--) t.removeRow(i); // clear
  FlexTable (except for first header row)
                if (!showInactive) {
                    showInactiveOps.setText("Skjul inaktive operatører");
                    showInactive = true;
                    getUsers();
```

```
76                  }
77                  else {
78                      showInactiveOps.setText("Vis inaktive operatører");
79                      showInactive = false;
80                      getUsers();
81                  }
82              }
83          });
84      }
85
86      private void getUsers() {
87          clientImpl.service.getUser(new AsyncCallback<List<UserDTO>>() {
88
89              @Override
90              public void onFailure(Throwable caught) {
91                  Window.alert("Server fejl!" + caught.getMessage());
92              }
93
94              @Override
95              public void onSuccess(List<UserDTO> result) {
96                  int j = 1;
97                  for (int i=0; i < result.size(); i++) {
98                      if (!showInactive) {
99                          if (Integer.valueOf(result.get(i).getActive()) == 1) {
100                             t.setText(j, 0, ""+result.get(i).getOprId());
101                             t.setText(j, 1, result.get(i).getOprNavn());
102                             t.setText(j, 2, result.get(i).getIni());
103                             t.setText(j, 3, result.get(i).getCpr());
104                             t.setText(j, 4, result.get(i).getPassword());
105                             t.setText(j, 5, result.get(i).getActive());
106                             t.setText(j, 6, result.get(i).getLevel());
107                             j++;
108                         }
109                     }
110                     else {
111                         t.setText(j, 0, ""+result.get(i).getOprId());
112                         t.setText(j, 1, result.get(i).getOprNavn());
113                         t.setText(j, 2, result.get(i).getIni());
114                         t.setText(j, 3, result.get(i).getCpr());
115                         t.setText(j, 4, result.get(i).getPassword());
116                         t.setText(j, 5, result.get(i).getActive());
117                         t.setText(j, 6, result.get(i).getLevel());
118                         j++;
119                     }
120                 }
121             }
122         });
123     }
124 }
125
```

```java
package dtu.client.ui;

import java.util.List;

public class UserDeleteView extends Composite {
    VerticalPanel deletePanel;
    FlexTable t;

    DatabaseServiceClientImpl clientImpl;

    // previous cancel anchor
    Anchor previousCancel = null;

    int eventRowIndex;

    public UserDeleteView(DatabaseServiceClientImpl clientImpl) {
        this.clientImpl = clientImpl;
        deletePanel = new VerticalPanel();
        initWidget(this.deletePanel);

        Label pageTitleLbl = new Label("Slet bruger");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.addStyleName("spacing-vertical");
        deletePanel.add(pageTitleLbl);

        t = new FlexTable();
        t.getFlexCellFormatter().setWidth(0, 0, "40px");
        t.getFlexCellFormatter().setWidth(0, 1, "150px");
        t.getFlexCellFormatter().setWidth(0, 2, "50px");
        t.getFlexCellFormatter().setWidth(0, 3, "110px");
        t.getFlexCellFormatter().setWidth(0, 4, "80px");
        t.getFlexCellFormatter().setWidth(0, 5, "40px");
        t.getFlexCellFormatter().setWidth(0, 6, "30px");

        t.addStyleName("FlexTable");
        t.getRowFormatter().addStyleName(0,"FlexTable-Header");

        // set headers in flextable
        t.setText(0, 0, "ID");
        t.setText(0, 1, "Navn");
        t.setText(0, 2, "Initialer");
        t.setText(0, 3, "CPR");
        t.setText(0, 4, "Password");
        t.setText(0, 5, "Aktiv");
        t.setText(0, 6, "Niveau");


        clientImpl.service.getUser(new AsyncCallback<List<UserDTO>>() {

            @Override
            public void onFailure(Throwable caught) {
                Window.alert("Server fejl!");
            }

            @Override
            public void onSuccess(List<UserDTO> result) {
                // populate table and add delete anchor to each row
                for (int i=0; i < result.size(); i++) {
                    if (Integer.valueOf(result.get(i).getActive()) == 1) {
                        t.setText(i+1, 0, ""+result.get(i).getOprId());
                        t.setText(i+1, 1, result.get(i).getOprNavn());
                        t.setText(i+1, 2, result.get(i).getIni());
```

```
76                          t.setText(i+1, 3, result.get(i).getCpr());
77                          t.setText(i+1, 4, result.get(i).getPassword());
78                          t.setText(i+1, 5, result.get(i).getActive());
79                          t.setText(i+1, 6, result.get(i).getLevel());
80                          Anchor delete = new Anchor("delete");
81                          t.setWidget(i+1, 7, delete);
82                          delete.addClickHandler(new DeleteHandler());
83                      }
84                  }
85              }
86          });
87
88          deletePanel.add(t);
89      }
90
91      private class DeleteHandler implements ClickHandler {
92          public void onClick(ClickEvent event) {
93
94              // if previous cancel open - force cancel operation
95              if (previousCancel != null)
96                  previousCancel.fireEvent(new ClickEvent(){});
97
98
99              // get rowindex where event happened
100             eventRowIndex = t.getCellForEvent(event).getRowIndex();
101
102             // get delete anchor ref for cancel operation
103             final Anchor delete =  (Anchor) event.getSource();
104             //          delete.setStyleName("delete-red");
105
106             Anchor ok = new Anchor("ok");
107             ok.addClickHandler(new ClickHandler() {
108
109                 @Override
110                 public void onClick(ClickEvent event) {
111
112                     // delete object with id in back end
113                     clientImpl.service.deleteUser(Integer.parseInt(t.getText(eventR
    owIndex, 0)), new AsyncCallback<Void>() {
114
115                         @Override
116                         public void onSuccess(Void result) {
117                             t.removeRow(eventRowIndex);
118                             //                          eventRowIndex-=1;
119                         }
120
121                         @Override
122                         public void onFailure(Throwable caught) {
123                             Window.alert("Server fejl!" + caught.getMessage());
124                         }
125                     });
126                     previousCancel = null;
127                 }
128             });
129
130             Anchor cancel = new Anchor("cancel");
131             previousCancel = cancel;
132             //          previousCancel.setStyleName("delete-red");
133             cancel.addClickHandler(new ClickHandler() {
134
135                 @Override
136                 public void onClick(ClickEvent event) {
```

```
137                     t.setWidget(eventRowIndex, 7, delete);
138                     t.clearCell(eventRowIndex, 8);
139                 }
140 
141             });
142 
143             // showing ok and cancel widgets
144             t.setWidget(eventRowIndex, 7, ok);
145             t.setWidget(eventRowIndex, 8, cancel);
146         }
147     }
148 }
149 
150 
151 
```

```java
package dtu.client.ui;

import java.util.List;


public class UserEditView extends Composite {
    VerticalPanel editPanel;
    FlexTable t;
    int rowCount;

    // editing text boxes
    TextBox nameTxt;
    TextBox iniTxt;
    TextBox cprTxt;
    TextBox passTxt;
    TextBox activeTxt;
    TextBox levelTxt;

    // valid fields - initially a field is valid
    boolean nameValid = true;
    boolean iniValid = true;
    boolean cprValid = true;
    boolean passValid = true;
    boolean activeValid = true;
    boolean levelValid = true;

    int eventRowIndex;

    DatabaseServiceClientImpl clientImpl;

    // user list
    List<UserDTO> brugere;

    // previous cancel anchor
    Anchor previousCancel = null;

    Button showInactiveOps;
    boolean showInactive = false;

    public UserEditView(DatabaseServiceClientImpl clientImpl) {
        this.clientImpl = clientImpl;

        editPanel = new VerticalPanel();
        initWidget(this.editPanel);

        HorizontalPanel topPanel = new HorizontalPanel();
        showInactiveOps = new Button("Vis inaktive brugere");
        Label pageTitleLbl = new Label("Ret brugere");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.setWidth("450px");
        topPanel.add(pageTitleLbl);
        topPanel.add(showInactiveOps);
        topPanel.addStyleName("spacing-vertical");
        editPanel.add(topPanel);

        t = new FlexTable();

        // adjust column widths
        t.getFlexCellFormatter().setWidth(0, 0, "40px");
        t.getFlexCellFormatter().setWidth(0, 1, "150px");
        t.getFlexCellFormatter().setWidth(0, 2, "50px");
        t.getFlexCellFormatter().setWidth(0, 3, "110px");
```

```
82          t.getFlexCellFormatter().setWidth(0, 4, "80px");
83          t.getFlexCellFormatter().setWidth(0, 5, "40px");
84          t.getFlexCellFormatter().setWidth(0, 6, "30px");
85
86          // style table
87          t.addStyleName("FlexTable");
88          t.getRowFormatter().addStyleName(0,"FlexTable-Header");
89
90          // set headers in flextable
91          t.setText(0, 0, "ID");
92          t.setText(0, 1, "Navn");
93          t.setText(0, 2, "Initialer");
94          t.setText(0, 3, "CPR");
95          t.setText(0, 4, "Password");
96          t.setText(0, 5, "Aktiv");
97          t.setText(0, 6, "Niveau");
98
99          getUsers();
100
101
102         editPanel.add(t);
103
104         // text boxes
105         nameTxt = new TextBox();
106         nameTxt.setWidth("140px");
107         iniTxt = new TextBox();
108         iniTxt.setWidth("40px");
109         cprTxt = new TextBox();
110         cprTxt.setWidth("100px");
111         passTxt = new TextBox();
112         passTxt.setWidth("70px");
113         activeTxt = new TextBox();
114         activeTxt.setWidth("20px");
115         levelTxt = new TextBox();
116         levelTxt.setWidth("20px");
117
118
119         showInactiveOps.addClickHandler(new ClickHandler() {
120
121             @Override
122             public void onClick(ClickEvent event) {
123                 // if previous edit open - force cancel operation
124                 if (previousCancel != null)
125                     previousCancel.fireEvent(new ClickEvent(){});
126                 rowCount = t.getRowCount();
127                 for (int i = rowCount-1; i > 0; i--) t.removeRow(i); // clear
    FlexTable (except for first header row)
128                 if (!showInactive) {
129                     showInactiveOps.setText("Skjul inaktive brugere");
130                     showInactive = true;
131                     getUsers();
132                 }
133                 else {
134                     showInactiveOps.setText("Vis inaktive brugere");
135                     showInactive = false;
136                     getUsers();
137                 }
138             }
139         });
140     }
141
142     private void getUsers() {
```

```
143         clientImpl.service.getUser(new AsyncCallback<List<UserDTO>>() {
144
145             @Override
146             public void onFailure(Throwable caught) {
147                 Window.alert("Server fejl! " + caught.getMessage());
148             }
149
150             @Override
151             public void onSuccess(List<UserDTO> result) {
152                 int j = 1;
153                 for (int rowIndex=0; rowIndex < result.size(); rowIndex++) {
154                     if (!showInactive) {
155                         if (Integer.valueOf(result.get(rowIndex).getActive()) == 1)
    {
156                             t.setText(j, 0, ""+result.get(rowIndex).getOprId());
157                             t.setText(j, 1, result.get(rowIndex).getOprNavn());
158                             t.setText(j, 2, result.get(rowIndex).getIni());
159                             t.setText(j, 3, result.get(rowIndex).getCpr());
160                             t.setText(j, 4, result.get(rowIndex).getPassword());
161                             t.setText(j, 5, result.get(rowIndex).getActive());
162                             t.setText(j, 6, result.get(rowIndex).getLevel());
163                             Anchor edit = new Anchor("edit");
164                             t.setWidget(j, 7, edit);
165                             edit.addClickHandler(new EditHandler());
166                             j++;
167                         }
168                     }
169                     else {
170                         t.setText(j, 0, ""+result.get(rowIndex).getOprId());
171                         t.setText(j, 1, result.get(rowIndex).getOprNavn());
172                         t.setText(j, 2, result.get(rowIndex).getIni());
173                         t.setText(j, 3, result.get(rowIndex).getCpr());
174                         t.setText(j, 4, result.get(rowIndex).getPassword());
175                         t.setText(j, 5, result.get(rowIndex).getActive());
176                         t.setText(j, 6, result.get(rowIndex).getLevel());
177                         Anchor edit = new Anchor("edit");
178                         t.setWidget(j, 7, edit);
179                         edit.addClickHandler(new EditHandler());
180                         j++;
181                     }
182                 }
183             }
184         });
185     }
186
187     private class EditHandler implements ClickHandler {
188         public void onClick(ClickEvent event) {
189
190             // if previous edit open - force cancel operation
191             if (previousCancel != null)
192                 previousCancel.fireEvent(new ClickEvent(){});
193
194             // get rowindex where event happened
195             eventRowIndex = t.getCellForEvent(event).getRowIndex();
196
197             // populate textboxes
198             nameTxt.setText(t.getText(eventRowIndex, 1));
199             iniTxt.setText(t.getText(eventRowIndex, 2));
200             cprTxt.setText(t.getText(eventRowIndex, 3));
201             passTxt.setText(t.getText(eventRowIndex, 4));
202             activeTxt.setText(t.getText(eventRowIndex, 5));
203             levelTxt.setText(t.getText(eventRowIndex, 6));
```

```
204
205             // show text boxes for editing
206             t.setWidget(eventRowIndex, 1, nameTxt);
207             t.setWidget(eventRowIndex, 2, iniTxt);
208             t.setWidget(eventRowIndex, 3, cprTxt);
209             t.setWidget(eventRowIndex, 4, passTxt);
210             t.setWidget(eventRowIndex, 5, activeTxt);
211             t.setWidget(eventRowIndex, 6, levelTxt);
212
213             // start editing here
214             nameTxt.setFocus(true);
215
216             // get edit anchor ref for cancel operation
217             final Anchor edit =  (Anchor) event.getSource();
218
219             // get textbox contents for cancel operation
220             final String name = nameTxt.getText();
221             final String ini = iniTxt.getText();
222             final String cpr = cprTxt.getText();
223             final String pass = passTxt.getText();
224             final String active = activeTxt.getText();
225             final String level = levelTxt.getText();
226
227
228             final Anchor ok = new Anchor("ok");
229             ok.addClickHandler(new ClickHandler() {
230
231                 @Override
232                 public void onClick(ClickEvent event) {
233
234                     // fill DTO with id and new values
235                     UserDTO userDTO = new
    UserDTO(Integer.parseInt(t.getText(eventRowIndex, 0)), nameTxt.getText(),
    iniTxt.getText(),
236                             cprTxt.getText(), passTxt.getText(),
    Integer.valueOf(activeTxt.getText()), Integer.valueOf(levelTxt.getText()));
237
238                     clientImpl.service.updateUser(userDTO, new
    AsyncCallback<Void>() {
239
240                         @Override
241                         public void onSuccess(Void result) {
242                             // remove inputboxes
243                             t.setText(eventRowIndex, 1, nameTxt.getText());
244                             t.setText(eventRowIndex, 2, iniTxt.getText());
245                             t.setText(eventRowIndex, 3, cprTxt.getText());
246                             t.setText(eventRowIndex, 4, passTxt.getText());
247                             t.setText(eventRowIndex, 5, activeTxt.getText());
248                             t.setText(eventRowIndex, 6, levelTxt.getText());
249                         }
250
251                         @Override
252                         public void onFailure(Throwable caught) {
253                             Window.alert("Server fejl!" + caught.getMessage());
254                         }
255
256                     });
257                     // restore edit link
258                     t.setWidget(eventRowIndex, 7, edit);
259                     t.clearCell(eventRowIndex, 8);
260
261                     previousCancel = null;
```

```
262                 }
263             });
264 
265             Anchor cancel = new Anchor("cancel");
266             previousCancel = cancel;
267             cancel.addClickHandler(new ClickHandler() {
268 
269                 @Override
270                 public void onClick(ClickEvent event) {
271 
272                     // restore original content of textboxes and rerun input validation
273 
274                     nameTxt.setText(name);
275                     nameTxt.fireEvent(new KeyUpEvent() {}); // validation
276 
277                     iniTxt.setText(ini);
278                     iniTxt.fireEvent(new KeyUpEvent() {});  // validation
279 
280                     cprTxt.setText(cpr);
281                     cprTxt.fireEvent(new KeyUpEvent() {}); // validation
282 
283                     passTxt.setText(pass);
284                     passTxt.fireEvent(new KeyUpEvent() {}); // validation
285 
286                     activeTxt.setText(pass);
287                     activeTxt.fireEvent(new KeyUpEvent() {}); // validation
288 
289                     levelTxt.setText(pass);
290                     levelTxt.fireEvent(new KeyUpEvent() {}); // validation
291 
292                     t.setText(eventRowIndex, 1, name);
293                     t.setText(eventRowIndex, 2, ini);
294                     t.setText(eventRowIndex, 3, cpr);
295                     t.setText(eventRowIndex, 4, pass);
296                     t.setText(eventRowIndex, 5, active);
297                     t.setText(eventRowIndex, 6, level);
298 
299                     // restore edit link
300                     t.setWidget(eventRowIndex, 7, edit);
301                     t.clearCell(eventRowIndex, 8);
302 
303                     previousCancel = null;
304                 }
305 
306             });
307 
308 
309             nameTxt.addKeyUpHandler(new KeyUpHandler(){
310 
311                 @Override
312                 public void onKeyUp(KeyUpEvent event) {
313                     if (!FieldVerifier.isValidName(nameTxt.getText())) {
314                         nameTxt.setStyleName("gwt-TextBox-invalidEntry");
315                         nameValid = false;
316                     }
317                     else {
318                         nameTxt.removeStyleName("gwt-TextBox-invalidEntry");
319                         nameValid = true;
320                     }
321 
322                     // enable/disable ok depending on form statusValid
```

```
323                     if
    (nameValid&&iniValid&&cprValid&&passValid&&activeValid&&levelValid)
324                         t.setWidget(eventRowIndex, 7, ok);
325                     else
326                         t.setText(eventRowIndex, 7, "ok");
327                 }
328
329             });
330
331             iniTxt.addKeyUpHandler(new KeyUpHandler(){
332
333                 @Override
334                 public void onKeyUp(KeyUpEvent event) {
335                     if (!FieldVerifier.isValidInitials(iniTxt.getText())) {
336                         iniTxt.setStyleName("gwt-TextBox-invalidEntry");
337                         iniValid = false;
338                     }
339                     else {
340                         iniTxt.removeStyleName("gwt-TextBox-invalidEntry");
341                         iniValid = true;
342                     }
343
344                     // enable/disable ok depending on form statusValid
345                     if
    (nameValid&&iniValid&&cprValid&&passValid&&activeValid&&levelValid)
346                         t.setWidget(eventRowIndex, 7, ok);
347                     else
348                         t.setText(eventRowIndex, 7, "ok");
349                 }
350
351             });
352
353             cprTxt.addKeyUpHandler(new KeyUpHandler(){
354
355                 @Override
356                 public void onKeyUp(KeyUpEvent event) {
357                     if (!FieldVerifier.isValidCpr(cprTxt.getText())) {
358                         cprTxt.setStyleName("gwt-TextBox-invalidEntry");
359                         cprValid = false;
360                     }
361                     else {
362                         cprTxt.removeStyleName("gwt-TextBox-invalidEntry");
363                         cprValid = true;
364                     }
365
366                     // enable/disable ok depending on form statusValid
367                     if
    (nameValid&&iniValid&&cprValid&&passValid&&activeValid&&levelValid)
368                         t.setWidget(eventRowIndex, 7, ok);
369                     else
370                         t.setText(eventRowIndex, 7, "ok");
371                 }
372
373             });
374
375             passTxt.addKeyUpHandler(new KeyUpHandler(){
376
377                 @Override
378                 public void onKeyUp(KeyUpEvent event) {
379                     if (!FieldVerifier.isValidPass(passTxt.getText())) {
380                         passTxt.setStyleName("gwt-TextBox-invalidEntry");
381                         passValid = false;
```

```
382                     }
383                     else {
384                         passTxt.removeStyleName("gwt-TextBox-invalidEntry");
385                         passValid = true;
386                     }
387 
388                     // enable/disable ok depending on form statusValid
389                     if
    (nameValid&&iniValid&&cprValid&&passValid&&activeValid&&levelValid)
390                         t.setWidget(eventRowIndex, 7, ok);
391                     else
392                         t.setText(eventRowIndex, 7, "ok");
393                 }
394 
395             });
396 
397             activeTxt.addKeyUpHandler(new KeyUpHandler(){
398 
399                 @Override
400                 public void onKeyUp(KeyUpEvent event) {
401                     if (!FieldVerifier.isValidActive(activeTxt.getText())) {
402                         activeTxt.setStyleName("gwt-TextBox-invalidEntry");
403                         activeValid = false;
404                     }
405                     else {
406                         activeTxt.removeStyleName("gwt-TextBox-invalidEntry");
407                         activeValid = true;
408                     }
409 
410                     // enable/disable ok depending on form statusValid
411                     if
    (nameValid&&iniValid&&cprValid&&passValid&&activeValid&&levelValid)
412                         t.setWidget(eventRowIndex, 7, ok);
413                     else
414                         t.setText(eventRowIndex, 7, "ok");
415                 }
416 
417             });
418 
419             levelTxt.addKeyUpHandler(new KeyUpHandler(){
420 
421                 @Override
422                 public void onKeyUp(KeyUpEvent event) {
423                     if (!FieldVerifier.isValidLevel(levelTxt.getText())) {
424                         levelTxt.setStyleName("gwt-TextBox-invalidEntry");
425                         levelValid = false;
426                     }
427                     else {
428                         levelTxt.removeStyleName("gwt-TextBox-invalidEntry");
429                         levelValid = true;
430                     }
431 
432                     // enable/disable ok depending on form statusValid
433                     if
    (nameValid&&iniValid&&cprValid&&passValid&&activeValid&&levelValid)
434                         t.setWidget(eventRowIndex, 7, ok);
435                     else
436                         t.setText(eventRowIndex, 7, "ok");
437                 }
438 
439             });
440 
```

```
441             // showing ok and cancel widgets
442             t.setWidget(eventRowIndex, 7, ok);
443             t.setWidget(eventRowIndex, 8, cancel);
444         }
445     }
446 }
447
```

```
 1 package dtu.client.ui;
 2
 3
 4 import com.google.gwt.event.dom.client.ClickEvent;
19
20
21 public class WelcomeView extends Composite {
22     DatabaseServiceClientImpl clientImpl;
23     TextBox userIdTxt;
24     TextBox passwordTxt;
25
26     public WelcomeView(final DatabaseServiceClientImpl clientImpl, final MainView
   mainView) {
27         this.clientImpl = clientImpl;
28         final VerticalPanel w = new VerticalPanel();
29         initWidget(w);
30         Label welcomeLbl = new Label("Velkommen til CDIO Webadministration,
   udviklet af CDIO gruppe 16.");
31         welcomeLbl.addStyleName("spacing-top");
32         w.add(welcomeLbl);
33
34         Label welcomeLbl2 = new Label("Du skal logge ind forneden for at komme
   videre.");
35         welcomeLbl2.addStyleName("spacing-bottom");
36         w.add(welcomeLbl2);
37
38         FlexTable loginTable = new FlexTable();
39
40         Label userIdLbl = new Label("ID:");
41         userIdTxt = new TextBox();
42         userIdTxt.setWidth("3em");
43         loginTable.setWidget(0, 0, userIdLbl);
44         loginTable.setWidget(0, 1, userIdTxt);
45
46         Label passwordLbl = new Label("Password:");
47         passwordTxt = new PasswordTextBox();
48         passwordTxt.setWidth("8em");
49         loginTable.setWidget(1, 0, passwordLbl);
50         loginTable.setWidget(1, 1, passwordTxt);
51
52         Button loginButton = new Button("Log ind");
53         loginTable.setWidget(2, 1, loginButton);
54
55         loginButton.addClickHandler(new ClickHandler() {
56             @Override
57             public void onClick(ClickEvent event) {
58                 clientImpl.service.login(Integer.valueOf(userIdTxt.getText()),
   passwordTxt.getText(), new AsyncCallback<Integer>() {
59
60                     @Override
61                     public void onSuccess(Integer result) {
62                         if (result==1) {
63
64                             // clear view and show new label
65                             Window.alert("Du er operatør og har ikke adgang til
   programmet");
66
67                         }
68                         else if(result==2) {
69                             mainView.showForemanMenu(result);
70
71                             // clear view and show new label
```

```
72                              w.clear();
73                              Label loginLbl = new Label("Du er nu logget ind og kan
   bruge navigationsmenuen foroven: " + result);
74                              loginLbl.addStyleName("spacing-vertical");
75                              w.add(loginLbl);
76
77                          }
78                          else if(result==3) {
79                              mainView.showpharmacistMenu(result);
80
81                              // clear view and show new label
82                              w.clear();
83                              Label loginLbl = new Label("Du er nu logget ind og kan
   bruge navigationsmenuen foroven: " + result);
84                              loginLbl.addStyleName("spacing-vertical");
85                              w.add(loginLbl);
86
87                          }
88                          else if(result==4) {
89                              mainView.showAdministratorMenu(result);
90
91                              // clear view and show new label
92                              w.clear();
93                              Label loginLbl = new Label("Du er nu logget ind og kan
   bruge navigationsmenuen foroven: " + result);
94                              loginLbl.addStyleName("spacing-vertical");
95                              w.add(loginLbl);
96
97                          }
98                          else  Window.alert("Ugyldigt login, prøv igen!");
99                      }
100
101                     @Override
102                     public void onFailure(Throwable caught) {
103                         Window.alert("Server feeeeeeeeeejl! " +
   caught.getMessage());
104                     }
105
106                 });
107             }
108         });
109
110         Label demoLabel = new Label("----- Demo knapper til login forneden -----");
111         demoLabel.addStyleName("redText");
112
113         Button SuperLogin = new Button("Log ind som superbruger");
114         SuperLogin.addStyleName("spacing-vertical");
115         SuperLogin.addClickHandler(new ClickHandler() {
116             @Override
117             public void onClick(ClickEvent event) {
118                 mainView.showAdministratorMenu(4);
119
120                 // clear view and show new label
121                 w.clear();
122                 Label loginLbl = new Label("Du er nu logget ind som superbruger og
   kan bruge navigationsmenuen foroven.");
123                 loginLbl.addStyleName("spacing-vertical");
124                 w.add(loginLbl);
125             }
126             });
127
128         Button Pharmaceutlogin = new Button("Log ind som farmaceut");
```

```
129         Pharmaceutlogin.addStyleName("spacing-vertical");
130         Pharmaceutlogin.addClickHandler(new ClickHandler() {
131             @Override
132             public void onClick(ClickEvent event) {
133                 mainView.showpharmacistMenu(3);
134 
135                 // clear view and show new label
136                 w.clear();
137                 Label loginLbl = new Label("Du er nu logget ind som farmaceut og
    kan bruge navigationsmenuen foroven.");
138                 loginLbl.addStyleName("spacing-vertical");
139                 w.add(loginLbl);
140             }
141             });
142 
143         Button Værkførelogin = new Button("Log ind som værkfører");
144         Værkførelogin.addStyleName("spacing-vertical");
145         Værkførelogin.addClickHandler(new ClickHandler() {
146             @Override
147             public void onClick(ClickEvent event) {
148                 mainView.showForemanMenu(2);
149 
150                 // clear view and show new label
151                 w.clear();
152                 Label loginLbl = new Label("Du er nu logget ind som værkfører og
    kan bruge navigationsmenuen foroven.");
153                 loginLbl.addStyleName("spacing-vertical");
154                 w.add(loginLbl);
155             }
156             });
157 
158         Button Operatørlogin = new Button("Log ind som Operatør");
159         Operatørlogin.addStyleName("spacing-vertical");
160         Operatørlogin.addClickHandler(new ClickHandler() {
161             @Override
162             public void onClick(ClickEvent event) {
163                 Window.alert("Du er operatør og har ikke adgang til
    webadministration.");
164             }
165             });
166 
167         loginTable.setWidget(3, 1, demoLabel);
168 
169         loginTable.setWidget(4, 1, SuperLogin);
170 
171         loginTable.setWidget(5, 1, Pharmaceutlogin);
172 
173         loginTable.setWidget(6, 1, Værkførelogin);
174 
175         loginTable.setWidget(7, 1, Operatørlogin);
176 
177         w.add(loginTable);
178     }
179 
180 }
181 
```

```java
package dtu.client.ui;

import com.google.gwt.event.dom.client.ClickEvent;

public class PharmacistMenu extends Composite {
    private HorizontalPanel hPanel = new HorizontalPanel();

    // receive reference to MainView for call back
    public PharmacistMenu(final MainView main, final int level) {
        initWidget(this.hPanel);

        Anchor raavare_administration = new Anchor("Råvare");
        hPanel.add(raavare_administration);
        raavare_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showCommoditiesMenu(level);
            }
        });

        Anchor raavarebatch_administration = new Anchor("Råvarebatch");
        hPanel.add(raavarebatch_administration);
        raavarebatch_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showCommoditiesBatchMenu(level);
            }
        });

        Anchor recept_administration = new Anchor("Recept");
        hPanel.add(recept_administration);
        recept_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showRecipeMenu(level);
            }
        });

        Anchor receptKomponent_administration = new Anchor("Receptkomponent");
        hPanel.add(receptKomponent_administration);
        receptKomponent_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showRecipeKomponentMenu(level);
            }
        });

        Anchor produktbatch_administration = new Anchor("Produktbatch");
        hPanel.add(produktbatch_administration);
        produktbatch_administration.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showProductBatchMenu(level);
            }
        });
    }
}

```

```java
package dtu.client.ui;


import com.google.gwt.user.client.Window;

public class ProductBatchAddView extends Composite {
    VerticalPanel addPanel;

    // controls
    Label produktBatchIdLbl;
    Label statusLbl;
    Label receptIdLbl;

    TextBox produktBatchIdTxt;
    TextBox statusTxt;
    TextBox receptIdTxt;

    Button save = new Button("Tilf\u00F8j produktbatch");

    // valid fields

    boolean prodBatchValid = false;
    boolean statusValid = false;
    boolean receptIdValid = false;

    public ProductBatchAddView(final DatabaseServiceClientImpl clientImpl) {

        addPanel = new VerticalPanel();
        initWidget(this.addPanel);

        FlexTable addTable = new FlexTable();

        Label pageTitleLbl = new Label("Tilføj produktbatch");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.addStyleName("spacing-vertical");
        addPanel.add(pageTitleLbl);

        produktBatchIdLbl = new Label("Produktbatch ID:");
        produktBatchIdTxt = new TextBox();
        Label nameRulesLbl = new Label("(Heltal kun)");
        addTable.setWidget(0, 0, produktBatchIdLbl);
        addTable.setWidget(0, 1, produktBatchIdTxt);
        addTable.setWidget(0, 2, nameRulesLbl);

        statusLbl = new Label("Status:");
        statusTxt = new TextBox();
        Label iniRulesLbl = new Label("(Heltal kun mellem 0 til 2)");
        addTable.setWidget(1, 0, statusLbl);
        addTable.setWidget(1, 1, statusTxt);
        addTable.setWidget(1, 2, iniRulesLbl);

        receptIdLbl = new Label("Recept ID:");
        receptIdTxt = new TextBox();
        Label passRulesLbl = new Label("(Heltal kun, skal findes i forvejen)");
        addTable.setWidget(2, 0, receptIdLbl);
        addTable.setWidget(2, 1, receptIdTxt);
        addTable.setWidget(2, 2, passRulesLbl);

        produktBatchIdTxt.setStyleName("gwt-TextBox-invalidEntry");
        statusTxt.setStyleName("gwt-TextBox-invalidEntry");
        receptIdTxt.setStyleName("gwt-TextBox-invalidEntry");

```

```
83          // use unicode escape sequence \u00F8 for '◆'
84          save.setEnabled(false);
85          addTable.setWidget(5, 1, save);
86
87          save.addClickHandler(new ClickHandler() {
88
89              @Override
90              public void onClick(ClickEvent event) {
91
92                  // create new OperatoerDTO
93
94                  ProductBatchDTO newProduktBatch= new
    ProductBatchDTO(Integer.valueOf(produktBatchIdTxt.getText()),
    Integer.valueOf(statusTxt.getText()), Integer.valueOf(receptIdTxt.getText()));
95
96                  // save on server
97                  clientImpl.service.createProductBatch(newProduktBatch, new
    AsyncCallback<Void>() {
98
99                      @Override
100                     public void onSuccess(Void result) {
101                         Window.alert("Produktbatch gemt i database.");
102                     }
103
104                     @Override
105                     public void onFailure(Throwable caught) {
106                         Window.alert("Server fejl!" + caught.getMessage());
107                     }
108
109                 });
110             }
111         });
112
113         produktBatchIdTxt.addKeyUpHandler(new KeyUpHandler(){
114
115             @Override
116             public void onKeyUp(KeyUpEvent event) {
117                 if (!FieldVerifier.isValidID(produktBatchIdTxt.getText())) {
118                     produktBatchIdTxt.setStyleName("gwt-TextBox-invalidEntry");
119                     prodBatchValid = false;
120                 }
121                 else {
122                     produktBatchIdTxt.removeStyleName("gwt-TextBox-invalidEntry");
123                     prodBatchValid = true;
124                 }
125                 checkFormValid();
126             }
127
128         });
129
130         statusTxt.addKeyUpHandler(new KeyUpHandler(){
131
132             @Override
133             public void onKeyUp(KeyUpEvent event) {
134                 if (!FieldVerifier.isValidStatus(statusTxt.getText())) {
135                     statusTxt.setStyleName("gwt-TextBox-invalidEntry");
136                     statusValid = false;
137                 }
138                 else {
139                     statusTxt.removeStyleName("gwt-TextBox-invalidEntry");
140                     statusValid = true;
141                 }
```

```
142                 checkFormValid();
143             }
144 
145         });
146 
147         receptIdTxt.addKeyUpHandler(new KeyUpHandler(){
148 
149             @Override
150             public void onKeyUp(KeyUpEvent event) {
151                 if (!FieldVerifier.IsValidMaengde(receptIdTxt.getText())) {
152                     receptIdTxt.setStyleName("gwt-TextBox-invalidEntry");
153                     receptIdValid = false;
154                 }
155                 else {
156                     receptIdTxt.removeStyleName("gwt-TextBox-invalidEntry");
157                     receptIdValid = true;
158                 }
159                 checkFormValid();
160             }
161 
162         });
163 
164 
165 
166 
167         addPanel.add(addTable);
168     }
169 
170     private void checkFormValid() {
171         if (statusValid&&receptIdValid&&prodBatchValid)
172             save.setEnabled(true);
173         else
174             save.setEnabled(false);
175     }
176 
177 }
178 
```

```java
package dtu.client.ui;

import java.util.List;

public class ProductBatchBrowseView extends Composite {
    DatabaseServiceClientImpl clientImpl;
    VerticalPanel browsePanel;
    FlexTable t;

    public ProductBatchBrowseView(DatabaseServiceClientImpl clientImpl) {
        this.clientImpl = clientImpl;
        browsePanel = new VerticalPanel();
        initWidget(this.browsePanel);

        HorizontalPanel topPanel = new HorizontalPanel();
        Label pageTitleLbl = new Label("Vis produktbatches");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.setWidth("450px");
        topPanel.add(pageTitleLbl);
        topPanel.addStyleName("spacing-vertical");
        browsePanel.add(topPanel);

        t = new FlexTable();

        t.getFlexCellFormatter().setWidth(0, 0, "8em");
        t.getFlexCellFormatter().setWidth(0, 1, "10em");
        t.getFlexCellFormatter().setWidth(0, 2, "6em");

        t.addStyleName("FlexTable");
        t.getRowFormatter().addStyleName(0,"FlexTable-Header");

        // set headers in flextable
        t.setText(0, 0, "Produktbatch ID");
        t.setText(0, 1, "Status");
        t.setText(0, 2, "Recept ID");

        getProduktBatches();

        browsePanel.add(t);
    }

    private void getProduktBatches() {
        clientImpl.service.getProductBatch(new
  AsyncCallback<List<ProductBatchDTO>>() {

            @Override
            public void onFailure(Throwable caught) {
                Window.alert("Server fejl!" + caught.getMessage());
            }

            @Override
            public void onSuccess(List<ProductBatchDTO> result) {
                for (int i=0; i < result.size(); i++) {
                    t.setText(i+1, 0, ""+result.get(i).getPb_ID());
                    t.setText(i+1, 1,""+ result.get(i).getStatus());
                    t.setText(i+1, 2,""+ result.get(i).getRecept_id());
                }
            }
        });
    }
}

```

```java
package dtu.shared;

import java.io.Serializable;

/**

 * Operatør Data Transfer Object

 */

public class ProductBatchDTO implements Serializable {

    /** Operatør id i området 1-99999999. Vælges af brugerne */
    int pb_ID;

    /** Operatør navn min. 2 max. 20 karakterer */
    int status;

    int recept_id;

    public ProductBatchDTO() {
    }

    public ProductBatchDTO(int pb_ID, int status, int recept_id) {
        this.pb_ID= pb_ID;
        this.recept_id = recept_id;
        this.status = status;
    }

    public int getPb_ID() {
        return pb_ID;
    }

    public void setPb_ID(int pb_ID) {
        this.pb_ID = pb_ID;
    }

    public int getStatus() {
        return status;
    }

    public void setStatus(int status) {
        this.status = status;
    }

    public int getRecept_id() {
        return recept_id;
    }

    public void setRecept_id(int recept_id) {
        this.recept_id = recept_id;
    }

}
```

```java
package dtu.client.ui;

import com.google.gwt.event.dom.client.ClickEvent;

public class ProductBatchMenu extends Composite {
    private HorizontalPanel hPanel = new HorizontalPanel();


    public ProductBatchMenu(final MainView main, final int level) {
        initWidget(this.hPanel);

        Anchor vis_ProduktBatch = new Anchor("Vis produktbatch");
        hPanel.add(vis_ProduktBatch);
        vis_ProduktBatch.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showProductBatch();
            }
        });

        Anchor opret_ProduktBatch = new Anchor("Opret produktbatch");
        hPanel.add(opret_ProduktBatch);
        opret_ProduktBatch.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.addProductBatch();
            }
        });

        Anchor back = new Anchor("Tilbage");
        hPanel.add(back);
        back.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.clearContentView();
                if (level == 4){
                    main.showAdministratorMenu(level);
                }
                else if (level == 3){
                    main.showpharmacistMenu(level);
                }

                else if (level == 2){
                    main.showForemanMenu(level);
                }
            }
        });
    }
}


```

```java
package dtu.shared;

import java.io.Serializable;

/**

 * Operatør Data Transfer Object

 */

public class ReceptKomponentDTO implements Serializable {

    /** Operatør id i området 1-99999999. Vælges af brugerne */
    int rcpId;

    /** Operatør navn min. 2 max. 20 karakterer */
    String rcpNavn;

    /** Operatør cpr-nr 10 karakterer */
    int rvrId;

    /** Operatør password min. 7 max. 8 karakterer */
    double nomNetto;

    /** Operatør aktiv (1) eller inaktiv (0) */
    double tolerance;

    public ReceptKomponentDTO(){

    }

    public ReceptKomponentDTO(int rcpId, int rvrId, double nomNetto, double tolerance) {
        this.rcpId = rcpId;
        this.rvrId = rvrId;
        this.nomNetto = nomNetto;
        this.tolerance = tolerance;
    }

    public int getRcpId() {
        return rcpId;
    }

    public void setRcpId(int rcpId) {
        this.rcpId = rcpId;
    }

    public int getRvrId() {
        return rvrId;
    }

    public void setRvrId(int rvrId) {
        this.rvrId = rvrId;
    }

    public double getNomNetto() {
        return nomNetto;
    }

    public void setNomNetto(double nomNetto) {
        this.nomNetto = nomNetto;
    }
```

```java

    public double getTolerance() {
        return tolerance;
    }

    public void setTolerance(double tolerance) {
        this.tolerance = tolerance;
    }

}
```

```java
package dtu.client.ui;


import com.google.gwt.user.client.Window;

public class RecipeAddView extends Composite {
    VerticalPanel addPanel;

    Label receptIdLbl;
    Label receptnameLbl;

    TextBox receptIdTxt;
    TextBox receptnameTxt;

    Button save = new Button("Tilf\u00F8j recept");

    // valid fields
    boolean recIDValid = false;
    boolean recNameValid = false;

    public RecipeAddView(final DatabaseServiceClientImpl clientImpl) {

        addPanel = new VerticalPanel();
        initWidget(this.addPanel);

        FlexTable addTable = new FlexTable();

        Label pageTitleLbl = new Label("Tilfoej recept");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.addStyleName("spacing-vertical");
        addPanel.add(pageTitleLbl);

        receptIdLbl = new Label("ID:");
        receptIdTxt = new TextBox();
        Label nameRulesLbl = new Label("(Heltal kun)");
        addTable.setWidget(0, 0, receptIdLbl);
        addTable.setWidget(0, 1, receptIdTxt);
        addTable.setWidget(0, 2, nameRulesLbl);

        receptnameLbl = new Label("Navn:");
        receptnameTxt = new TextBox();
        Label receptnameRulesLbl = new Label("(Mellem 2 og 20 karakterer)");
        addTable.setWidget(1, 0, receptnameLbl);
        addTable.setWidget(1, 1, receptnameTxt);
        addTable.setWidget(1, 2, receptnameRulesLbl);

        receptIdTxt.setStyleName("gwt-TextBox-invalidEntry");
        receptnameTxt.setStyleName("gwt-TextBox-invalidEntry");

        // use unicode escape sequence \u00F8 for '\u00f8'
        save.setEnabled(false);
        addTable.setWidget(5, 1, save);

        save.addClickHandler(new ClickHandler() {

            @Override
            public void onClick(ClickEvent event) {

                // create new OperatoerDTO

                RecipeDTO newRecept = new
  RecipeDTO(Integer.valueOf(receptIdTxt.getText()), receptnameTxt.getText());
```

```
80
81                  // save on server
82                  clientImpl.service.createRecipe(newRecept, new
  AsyncCallback<Void>() {
83
84                      @Override
85                      public void onSuccess(Void result) {
86                          Window.alert("Recept gemt i database.");
87                      }
88
89                      @Override
90                      public void onFailure(Throwable caught) {
91                          Window.alert("Server fejl!" + caught.getMessage());
92                      }
93
94                  });
95              }
96          });
97
98
99          // register event handlers
100
101         receptIdTxt.addKeyUpHandler(new KeyUpHandler(){
102
103             @Override
104             public void onKeyUp(KeyUpEvent event) {
105                 if (!FieldVerifier.isValidID(receptIdTxt.getText())) {
106                     receptIdTxt.setStyleName("gwt-TextBox-invalidEntry");
107                     recIDValid = false;
108                 }
109                 else {
110                     receptIdTxt.removeStyleName("gwt-TextBox-invalidEntry");
111                     recIDValid = true;
112                 }
113                 checkFormValid();
114             }
115
116         });
117
118         receptnameTxt.addKeyUpHandler(new KeyUpHandler(){
119
120             @Override
121             public void onKeyUp(KeyUpEvent event) {
122                 if (!FieldVerifier.isValidName(receptnameTxt.getText())) {
123                     receptnameTxt.setStyleName("gwt-TextBox-invalidEntry");
124                     recNameValid = false;
125                 }
126                 else {
127                     receptnameTxt.removeStyleName("gwt-TextBox-invalidEntry");
128                     recNameValid = true;
129                 }
130                 checkFormValid();
131             }
132
133         });
134         addPanel.add(addTable);
135     }
136
137     private void checkFormValid() {
138         if (recIDValid&&recNameValid)
139             save.setEnabled(true);
140         else
```

```
141             save.setEnabled(false);
142     }
143
144
145 }
146
```

```java
package dtu.client.ui;

import java.util.List;

public class RecipeBrowseView extends Composite {
    DatabaseServiceClientImpl clientImpl;
    VerticalPanel browsePanel;
    FlexTable t;

    public RecipeBrowseView(DatabaseServiceClientImpl clientImpl) {
        this.clientImpl = clientImpl;
        browsePanel = new VerticalPanel();
        initWidget(this.browsePanel);

        HorizontalPanel topPanel = new HorizontalPanel();
//      showInactiveOps = new Button("Vis inaktive operatører");
        Label pageTitleLbl = new Label("Vis recepter");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.setWidth("450px");
        topPanel.add(pageTitleLbl);
//      topPanel.add(showInactiveOps);
        topPanel.addStyleName("spacing-vertical");
        browsePanel.add(topPanel);

        t = new FlexTable();

        t.getFlexCellFormatter().setWidth(0, 0, "8em");
        t.getFlexCellFormatter().setWidth(0, 1, "40em");

        t.addStyleName("FlexTable");
        t.getRowFormatter().addStyleName(0,"FlexTable-Header");

        // set headers in flextable
        t.setText(0, 0, "ID");
        t.setText(0, 1, "Navn");

        getRecepter();

        browsePanel.add(t);
    }

    private void getRecepter() {
        clientImpl.service.getRecipe(new AsyncCallback<List<RecipeDTO>>() {

            @Override
            public void onFailure(Throwable caught) {
                Window.alert("Server fejl!" + caught.getMessage());
            }

            @Override
            public void onSuccess(List<RecipeDTO> result) {
                for (int i=0; i < result.size(); i++) {
                    t.setText(i+1, 0, ""+result.get(i).getRcpId());
                    t.setText(i+1, 1, ""+ result.get(i).getRecept_Navn());

                }
            }
        });
    }
}

```

```java
package dtu.client.ui;


import com.google.gwt.user.client.Window;

public class RecipeComponentAddView extends Composite {
    VerticalPanel addPanel;

    // controls
    Label recKompIdLbl;
    Label raavareidLbl;
    Label nomNetto;
    Label tolerance;

    TextBox recKompIdTxt;
    TextBox raavareIdTxt;
    TextBox nomNettoTxt;
    TextBox toleranceTxt;

    Button save = new Button("Tilf\u00F8j receptkomponent");

    // valid fields
    boolean rk_idValid = false;
    boolean raav_idValid = false;
    boolean nettoValid = false;
    boolean tolValid = false;

    public RecipeComponentAddView(final DatabaseServiceClientImpl clientImpl) {

        addPanel = new VerticalPanel();
        initWidget(this.addPanel);

        FlexTable addTable = new FlexTable();

        Label pageTitleLbl = new Label("Tilfoej receptkomponent");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.addStyleName("spacing-vertical");
        addPanel.add(pageTitleLbl);

        recKompIdLbl = new Label("Recept ID:");
        recKompIdTxt = new TextBox();
        Label nameRulesLbl = new Label("(Heltal kun, skal findes i forvejen)");
        addTable.setWidget(0, 0, recKompIdLbl);
        addTable.setWidget(0, 1, recKompIdTxt);
        addTable.setWidget(0, 2, nameRulesLbl);

        raavareidLbl = new Label("Råvare ID:");
        raavareIdTxt = new TextBox();
        Label iniRulesLbl = new Label("(Heltal kun, skal findes i forvejen)");
        addTable.setWidget(1, 0, raavareidLbl);
        addTable.setWidget(1, 1, raavareIdTxt);
        addTable.setWidget(1, 2, iniRulesLbl);

        nomNetto = new Label("Nom. netto:");
        nomNettoTxt = new TextBox();
        Label passRulesLbl = new Label("(Decimaltal mellem 0.1% - 10.0%");
        addTable.setWidget(3, 0, nomNetto);
        addTable.setWidget(3, 1, nomNettoTxt);
        addTable.setWidget(3, 2, passRulesLbl);

        tolerance = new Label("Tolerance:");
        toleranceTxt = new TextBox();
```

```
 81         Label activeRulesLbl = new Label("(Decimaltal mellem 0.1% - 10.0%)");
 82         addTable.setWidget(4, 0, tolerance);
 83         addTable.setWidget(4, 1, toleranceTxt);
 84         addTable.setWidget(4, 2, activeRulesLbl);
 85 
 86 
 87         recKompIdTxt.setStyleName("gwt-TextBox-invalidEntry");
 88         raavareIdTxt.setStyleName("gwt-TextBox-invalidEntry");
 89         nomNettoTxt.setStyleName("gwt-TextBox-invalidEntry");
 90         toleranceTxt.setStyleName("gwt-TextBox-invalidEntry");
 91 
 92         // use unicode escape sequence \u00F8 for 'ø'
 93         save.setEnabled(false);
 94         addTable.setWidget(5, 1, save);
 95 
 96         save.addClickHandler(new ClickHandler() {
 97 
 98             @Override
 99             public void onClick(ClickEvent event) {
100 
101                 // create new OperatoerDTO
102 
103                 ReceptKomponentDTO newReceptKomponent = new
    ReceptKomponentDTO(Integer.valueOf(recKompIdTxt.getText()),
    Integer.valueOf(raavareIdTxt.getText()), Double.valueOf(nomNettoTxt.getText()) ,
    Double.valueOf(toleranceTxt.getText()));
104 
105                 // save on server
106                 clientImpl.service.createRecipeComponent(newReceptKomponent, new
    AsyncCallback<Void>() {
107 
108                     @Override
109                     public void onSuccess(Void result) {
110                         Window.alert("Receptkomponent gemt i database.");
111                     }
112 
113                     @Override
114                     public void onFailure(Throwable caught) {
115                         Window.alert("Server fejl!" + caught.getMessage());
116                     }
117 
118                 });
119             }
120         });
121 
122 
123         // register event handlers
124 
125         recKompIdTxt.addKeyUpHandler(new KeyUpHandler(){
126 
127             @Override
128             public void onKeyUp(KeyUpEvent event) {
129                 if (!FieldVerifier.isValidID(recKompIdTxt.getText())) {
130                     recKompIdTxt.setStyleName("gwt-TextBox-invalidEntry");
131                     raav_idValid = false;
132                 }
133                 else {
134                     recKompIdTxt.removeStyleName("gwt-TextBox-invalidEntry");
135                     raav_idValid = true;
136                 }
137                 checkFormValid();
138             }
```

```
139
140         });
141
142         raavareIdTxt.addKeyUpHandler(new KeyUpHandler(){
143
144             @Override
145             public void onKeyUp(KeyUpEvent event) {
146                 if (!FieldVerifier.isValidID(raavareIdTxt.getText())) {
147                     raavareIdTxt.setStyleName("gwt-TextBox-invalidEntry");
148                     rk_idValid = false;
149                 }
150                 else {
151                     raavareIdTxt.removeStyleName("gwt-TextBox-invalidEntry");
152                     rk_idValid = true;
153                 }
154                 checkFormValid();
155             }
156
157         });
158
159         nomNettoTxt.addKeyUpHandler(new KeyUpHandler(){
160
161             @Override
162             public void onKeyUp(KeyUpEvent event) {
163                 if (!FieldVerifier.isValidNomNetto(nomNettoTxt.getText())) {
164                     nomNettoTxt.setStyleName("gwt-TextBox-invalidEntry");
165                     nettoValid = false;
166                 }
167                 else {
168                     nomNettoTxt.removeStyleName("gwt-TextBox-invalidEntry");
169                     nettoValid = true;
170                 }
171                 checkFormValid();
172             }
173
174         });
175
176         toleranceTxt.addKeyUpHandler(new KeyUpHandler(){
177
178             @Override
179             public void onKeyUp(KeyUpEvent event) {
180                 if (!FieldVerifier.isValidTolerance(toleranceTxt.getText())) {
181                     toleranceTxt.setStyleName("gwt-TextBox-invalidEntry");
182                     tolValid = false;
183                 }
184                 else {
185                     toleranceTxt.removeStyleName("gwt-TextBox-invalidEntry");
186                     tolValid = true;
187                 }
188                 checkFormValid();
189             }
190
191         });
192
193
194         addPanel.add(addTable);
195     }
196
197     private void checkFormValid() {
198         if (rk_idValid&&raav_idValid&&nettoValid&&tolValid)
199             save.setEnabled(true);
200         else
```

```
201             save.setEnabled(false);
202     }
203
204 }
205
```

```java
package dtu.client.ui;

import java.util.List;

public class RecipeComponentBrowseView extends Composite {
    DatabaseServiceClientImpl clientImpl;
    VerticalPanel browsePanel;
    FlexTable t;

    public RecipeComponentBrowseView(DatabaseServiceClientImpl clientImpl) {
        this.clientImpl = clientImpl;
        browsePanel = new VerticalPanel();
        initWidget(this.browsePanel);

        HorizontalPanel topPanel = new HorizontalPanel();
        Label pageTitleLbl = new Label("Vis recepter");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.setWidth("450px");
        topPanel.add(pageTitleLbl);
        topPanel.addStyleName("spacing-vertical");
        browsePanel.add(topPanel);

        t = new FlexTable();

        t.getFlexCellFormatter().setWidth(0, 0, "40px");
        t.getFlexCellFormatter().setWidth(0, 1, "40px");
        t.getFlexCellFormatter().setWidth(0, 2, "60px");
        t.getFlexCellFormatter().setWidth(0, 3, "50px");

        t.addStyleName("FlexTable");
        t.getRowFormatter().addStyleName(0,"FlexTable-Header");

        // set headers in flextable
        t.setText(0, 0, "Recept ID");
        t.setText(0, 1, "Raavare ID");
        t.setText(0, 2, "Nom. Netto");
        t.setText(0, 3, "Tolerance");

        getReceptKomponenter();
        browsePanel.add(t);
    }

    private void getReceptKomponenter() {
        clientImpl.service.getRecipeComponent(new
  AsyncCallback<List<ReceptKomponentDTO>>() {

            @Override
            public void onFailure(Throwable caught) {
                Window.alert("Server fejl!" + caught.getMessage());
            }

            @Override
            public void onSuccess(List<ReceptKomponentDTO> result) {
                for (int i=0; i < result.size(); i++) {
                    t.setText(i+1, 0, ""+result.get(i).getRcpId());
                    t.setText(i+1, 1,""+ result.get(i).getRvrId());
                    t.setText(i+1, 2,""+ result.get(i).getNomNetto());
                    t.setText(i+1, 3,""+ result.get(i).getTolerance());

                }
            }
        });
```

```
78     }
79 }
80
```

```java
package dtu.client.ui;

import com.google.gwt.event.dom.client.ClickEvent;

public class RecipeComponentMenu extends Composite {
    private HorizontalPanel hPanel = new HorizontalPanel();


    public RecipeComponentMenu(final MainView main, final int level) {
        initWidget(this.hPanel);

        Anchor vis_receptkomp = new Anchor("Vis receptomponenter");
        hPanel.add(vis_receptkomp);
        vis_receptkomp.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showRecipeComponents();
            }
        });

        Anchor tilfoej_receptkomponent = new Anchor("Tilføj receptkomponent");
        hPanel.add(tilfoej_receptkomponent);
        tilfoej_receptkomponent.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.addRecipeComponent();
            }
        });

        Anchor back = new Anchor("Tilbage");
        hPanel.add(back);
        back.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.clearContentView();
                if (level == 4){
                    main.showAdministratorMenu(level);
                }
                else if (level == 3){
                    main.showpharmacistMenu(level);
                }

                else if (level == 2){
                    main.showForemanMenu(level);
                }
            }
        });
    }
}
```

```java
package dtu.shared;

import java.io.Serializable;

/**

 * Operatør Data Transfer Object

 */

public class RecipeDTO implements Serializable {

    /** Operatør id i området 1-99999999. Vælges af brugerne */
    int recept_ID;

    /** Operatør navn min. 2 max. 20 karakterer */
    String recept_Navn;

    public RecipeDTO() {
    }

    public RecipeDTO(int recept_ID, String raavare_Navn) {
        this.recept_ID= recept_ID;
        this.recept_Navn = raavare_Navn;
    }

    public void setRecept_ID(int recept_ID) {
        this.recept_ID = recept_ID;
    }

    public String getRecept_Navn() {
        return recept_Navn;
    }

    public void setRecept_Navn(String recept_Navn) {
        this.recept_Navn = recept_Navn;
    }

    public int getRcpId() {
        return recept_ID;
    }



}
```

```java
package dtu.client.ui;


import com.google.gwt.user.client.Window;

public class UserAddView extends Composite {
    VerticalPanel addPanel;

    // controls
    Label nameLbl;
    Label iniLbl;
    Label cprLbl;
    Label passLbl;
    Label activeLbl;
    Label levelLbl;

    TextBox nameTxt;
    TextBox iniTxt;
    TextBox cprTxt;
    TextBox passTxt;
    TextBox activeTxt;
    TextBox levelTxt;

    Button save = new Button("Tilf\u00F8j bruger");

    // valid fields
    boolean nameValid = false;
    boolean iniValid = false;
    boolean cprValid = false;
    boolean passValid = false;
    boolean activeValid = false;
    boolean levelValid = false;

    public UserAddView(final DatabaseServiceClientImpl clientImpl) {

        addPanel = new VerticalPanel();
        initWidget(this.addPanel);

        FlexTable addTable = new FlexTable();

        Label pageTitleLbl = new Label("Tilføj bruger");
        pageTitleLbl.setStyleName("FlexTable-Header");
        pageTitleLbl.addStyleName("spacing-vertical");
        addPanel.add(pageTitleLbl);

        nameLbl = new Label("Navn:");
        nameTxt = new TextBox();
        Label nameRulesLbl = new Label("(min. 2 og max. 20 karakterer)");
        addTable.setWidget(0, 0, nameLbl);
        addTable.setWidget(0, 1, nameTxt);
        addTable.setWidget(0, 2, nameRulesLbl);

        iniLbl = new Label("Initialer:");
        iniTxt = new TextBox();
        Label iniRulesLbl = new Label("(min. 2 og max. 3 karakterer)");
        addTable.setWidget(1, 0, iniLbl);
        addTable.setWidget(1, 1, iniTxt);
        addTable.setWidget(1, 2, iniRulesLbl);

        cprLbl = new Label("CPR:");
        cprTxt = new TextBox();
        Label cprRulesLbl = new Label("(10 karakterer)");
```

```
 79         addTable.setWidget(2, 0, cprLbl);
 80         addTable.setWidget(2, 1, cprTxt);
 81         addTable.setWidget(2, 2, cprRulesLbl);
 82 
 83         passLbl = new Label("Password:");
 84         passTxt = new TextBox();
 85         Label passRulesLbl = new Label("(7-8 karakterer og skal overholde DTU's
    regler for password.)");
 86         addTable.setWidget(3, 0, passLbl);
 87         addTable.setWidget(3, 1, passTxt);
 88         addTable.setWidget(3, 2, passRulesLbl);
 89 
 90         activeLbl = new Label("Aktiv:");
 91         activeTxt = new TextBox();
 92         Label activeRulesLbl = new Label("(1 = aktiv, 0 = inaktiv)");
 93         addTable.setWidget(4, 0, activeLbl);
 94         addTable.setWidget(4, 1, activeTxt);
 95         addTable.setWidget(4, 2, activeRulesLbl);
 96 
 97         levelLbl = new Label("Niveau:");
 98         levelTxt = new TextBox();
 99         Label levelRulesLbl = new Label("(1 = operatør, 2 = værkfører, 3 =
    farmaceut, 4 = superbruger)");
100         addTable.setWidget(5, 0, levelLbl);
101         addTable.setWidget(5, 1, levelTxt);
102         addTable.setWidget(5, 2, levelRulesLbl);
103 
104         nameTxt.setStyleName("gwt-TextBox-invalidEntry");
105         iniTxt.setStyleName("gwt-TextBox-invalidEntry");
106         cprTxt.setStyleName("gwt-TextBox-invalidEntry");
107         passTxt.setStyleName("gwt-TextBox-invalidEntry");
108         activeTxt.setStyleName("gwt-TextBox-invalidEntry");
109         levelTxt.setStyleName("gwt-TextBox-invalidEntry");
110 
111         // use unicode escape sequence \u00F8 for 'ø'
112         save = new Button("Tilf\u00F8j");
113         save.setEnabled(false);
114         addTable.setWidget(6, 1, save);
115 
116         save.addClickHandler(new ClickHandler() {
117 
118             @Override
119             public void onClick(ClickEvent event) {
120 
121                 UserDTO newUser = new UserDTO(nameTxt.getText(), iniTxt.getText(),
    cprTxt.getText(), passTxt.getText(), Integer.valueOf(activeTxt.getText()),
    Integer.valueOf(levelTxt.getText()));
122 
123                 // save on server
124                 clientImpl.service.createUser(newUser, new AsyncCallback<Void>() {
125 
126                     @Override
127                     public void onSuccess(Void result) {
128                         Window.alert("Bruger gemt i database.");
129                     }
130 
131                     @Override
132                     public void onFailure(Throwable caught) {
133                         Window.alert("Server fejl!" + caught.getMessage());
134                     }
135 
136                 });
```

```
137             }
138         });
139 
140 
141         // register event handlers
142 
143         nameTxt.addKeyUpHandler(new KeyUpHandler(){
144 
145             @Override
146             public void onKeyUp(KeyUpEvent event) {
147                 if (!FieldVerifier.isValidName(nameTxt.getText())) {
148                     nameTxt.setStyleName("gwt-TextBox-invalidEntry");
149                     nameValid = false;
150                 }
151                 else {
152                     nameTxt.removeStyleName("gwt-TextBox-invalidEntry");
153                     nameValid = true;
154                 }
155                 checkFormValid();
156             }
157 
158         });
159 
160         iniTxt.addKeyUpHandler(new KeyUpHandler(){
161 
162             @Override
163             public void onKeyUp(KeyUpEvent event) {
164                 if (!FieldVerifier.isValidInitials(iniTxt.getText())) {
165                     iniTxt.setStyleName("gwt-TextBox-invalidEntry");
166                     iniValid = false;
167                 }
168                 else {
169                     iniTxt.removeStyleName("gwt-TextBox-invalidEntry");
170                     iniValid = true;
171                 }
172                 checkFormValid();
173             }
174 
175         });
176 
177         cprTxt.addKeyUpHandler(new KeyUpHandler(){
178 
179             @Override
180             public void onKeyUp(KeyUpEvent event) {
181                 if (!FieldVerifier.isValidCpr(cprTxt.getText())) {
182                     cprTxt.setStyleName("gwt-TextBox-invalidEntry");
183                     cprValid = false;
184                 }
185                 else {
186                     cprTxt.removeStyleName("gwt-TextBox-invalidEntry");
187                     cprValid = true;
188                 }
189                 checkFormValid();
190             }
191 
192         });
193 
194         passTxt.addKeyUpHandler(new KeyUpHandler(){
195 
196             @Override
197             public void onKeyUp(KeyUpEvent event) {
198                 if (!FieldVerifier.isValidPass(passTxt.getText())) {
```

```
199                     passTxt.setStyleName("gwt-TextBox-invalidEntry");
200                     passValid = false;
201                 }
202                 else {
203                     passTxt.removeStyleName("gwt-TextBox-invalidEntry");
204                     passValid = true;
205                 }
206                 checkFormValid();
207             }
208 
209         });
210 
211         activeTxt.addKeyUpHandler(new KeyUpHandler(){
212 
213             @Override
214             public void onKeyUp(KeyUpEvent event) {
215                 if (!FieldVerifier.isValidActive(activeTxt.getText())) {
216                     activeTxt.setStyleName("gwt-TextBox-invalidEntry");
217                     activeValid = false;
218                 }
219                 else {
220                     activeTxt.removeStyleName("gwt-TextBox-invalidEntry");
221                     activeValid = true;
222                 }
223                 checkFormValid();
224             }
225 
226         });
227 
228         levelTxt.addKeyUpHandler(new KeyUpHandler(){
229 
230             @Override
231             public void onKeyUp(KeyUpEvent event) {
232                 if (!FieldVerifier.isValidLevel(levelTxt.getText())) {
233                     levelTxt.setStyleName("gwt-TextBox-invalidEntry");
234                     levelValid = false;
235                 }
236                 else {
237                     levelTxt.removeStyleName("gwt-TextBox-invalidEntry");
238                     levelValid = true;
239                 }
240                 checkFormValid();
241             }
242 
243         });
244 
245         addPanel.add(addTable);
246     }
247 
248     private void checkFormValid() {
249         if (nameValid&&iniValid&&cprValid&&passValid&&activeValid&&levelValid)
250             save.setEnabled(true);
251         else
252             save.setEnabled(false);
253     }
254 
255 }
256 
```

```java
package dtu.client.ui;

import com.google.gwt.event.dom.client.ClickEvent;

public class UserAdminMenu extends Composite {
    private HorizontalPanel hPanel = new HorizontalPanel();

    // receive reference to MainView for call back
    public UserAdminMenu(final MainView main, final int level) {
        initWidget(this.hPanel);

        Anchor showUsers = new Anchor("Vis brugere");
        hPanel.add(showUsers);

        // call back the controller
        showUsers.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.showUsers();
            }
        });

        // use unicode escape sequence \u00F8 for '�'
        Anchor add = new Anchor("Tilf\u00F8j brugere");
        hPanel.add(add);
        add.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.addUser();
            }
        });

        Anchor edit = new Anchor("Ret brugere");
        hPanel.add(edit);
        edit.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.editUsers();
            }
        });

        Anchor delete = new Anchor("Slet brugere");
        hPanel.add(delete);
        delete.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.deleteUsers();
            }
        });

        Anchor back = new Anchor("Tilbage");
        hPanel.add(back);
        back.addClickHandler(new ClickHandler(){
            public void onClick(ClickEvent event){
                main.clearContentView();
                if(level==4){
                    main.showAdministratorMenu(level);
                }
                else if(level == 3){
                    main.showpharmacistMenu(level);
                }

                else if(level == 2){
                    main.showForemanMenu(level);
                }
        }
```

```
69     });
70 }
71 }
72
```

```java
package dtu.shared;

import java.io.Serializable;

/**

 * Operatør Data Transfer Object

 */

public class UserDTO implements Serializable {

    /** Operatør id i området 1-99999999. Vælges af brugerne */
    int oprId;

    /** Operatør navn min. 2 max. 20 karakterer */
    String oprNavn;

    /** Operatør initialer min. 2 max. 3 karakterer */
    String ini;

    /** Operatør cpr-nr 10 karakterer */
    String cpr;

    /** Operatør password min. 7 max. 8 karakterer */
    String password;

    /** Operatør aktiv (1) eller inaktiv (0) */
    int active;

    /** Operatør niveau - 1 = operatør, 2 = superbruger */
    int level;

    public UserDTO() {
    }

    public UserDTO(String oprNavn, String ini, String cpr, String password, int active, int level) {
        this.oprNavn = oprNavn;
        this.ini = ini;
        this.cpr = cpr;
        this.password = password;
        this.active = active;
        this.level = level;
    }

    public UserDTO(int oprId, String oprNavn, String ini, String cpr, String password, int active, int level) {
        this.oprId = oprId;
        this.oprNavn = oprNavn;
        this.ini = ini;
        this.cpr = cpr;
        this.password = password;
        this.active = active;
        this.level = level;
    }

    public int getOprId() {
        return oprId;
    }

    public void setOprId(int oprId) {
```

```
61          this.oprId = oprId;
62      }
63
64      public String getOprNavn() {
65          return oprNavn;
66      }
67
68      public void setOprNavn(String oprNavn) {
69          this.oprNavn = oprNavn;
70      }
71
72      public String getIni() {
73          return ini;
74      }
75
76      public void setIni(String ini) {
77          this.ini = ini;
78      }
79
80      public String getCpr() {
81          return cpr;
82      }
83
84      public void setCpr(String cpr) {
85          this.cpr = cpr;
86      }
87
88      public String getPassword() {
89          return password;
90      }
91
92      public void setPassword(String password) {
93          this.password = password;
94      }
95
96      public String getActive() {
97          return String.valueOf(active);
98      }
99
100     public void setActive(int active) {
101         this.active = active;
102     }
103
104     public String getLevel() {
105         return String.valueOf(level);
106     }
107
108     public void setLevel(int level) {
109         this.level = level;
110     }
111
112 }
```

Bilag 16 - ASE kildekode

```java
package ASE;
import java.sql.*;

public class DAL {

    static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";

    static final String DB_URL = "jdbc:mysql://62.79.16.16/grp16";
    static final String USER = "grp16";
    static final String PASS = "ZHnPq74Y";
    Connection conn;
    Statement stmt;

    public DAL() {
        try {

            Class.forName("com.mysql.jdbc.Driver");

//          System.out.println("Connecting to database...");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);

            stmt = conn.createStatement();


        } catch (SQLException se) {
            se.printStackTrace();

        }
        catch (Exception e) {
            e.printStackTrace();

        }

    }

    public void insertProduktBatchKomp(int pb_id, int rb_id, double tara, double netto, int opr_id){
        try {
            String sql = String.format(Locale.US, "INSERT INTO produktbatchkomponent VALUES (%d, %d, %f, %f, %d)", pb_id, rb_id, tara, netto, opr_id);
            stmt = conn.createStatement();
            int result = stmt.executeUpdate(sql);
        } catch(SQLException e) {
            System.out.println(e.getMessage());
        }
    }

    public String setProduktBatchStatus(int pb_id, int status){
        try {
            String sql = "UPDATE produktbatch SET status = "+status+" WHERE pb_id = "+pb_id;
            stmt = conn.createStatement();
            int result = stmt.executeUpdate(sql);
            return "Success";
        } catch(SQLException e) {
            return "SQL Fejl";
        }
    }

    public String getReceptNavnFromPBID(String id){
        try {
```

```
63              String sql = "SELECT recept_navn FROM recept WHERE recept_id = (SELECT
  recept_id FROM produktbatch WHERE pb_id = " + id + ");";
64              stmt = conn.createStatement();
65              ResultSet rs = stmt.executeQuery(sql);
66
67              if (rs.next()) {
68                  return rs.getString("recept_navn");
69              } else {
70                  return "ID findes ikke!";
71              }
72          } catch(SQLException e) {
73              return "SQL Fejl";
74          }
75      }
76
77      public String getRaavarebatch(int id, int raav_id){
78          try {
79              String sql = "SELECT raavare_id FROM raavarebatch WHERE rb_id = "+id;
80              stmt = conn.createStatement();
81              ResultSet rs = stmt.executeQuery(sql);
82
83              if (rs.next()) {
84                  if (raav_id == rs.getInt("raavare_id")) {
85                      return String.valueOf(id);
86                  }
87              }
88              return "ID findes ikke!";
89
90              } catch(SQLException e) {
91                  return "SQL Fejl";
92              }
93          }
94
95          public String getRaavareNameFromID(int id){
96              try {
97                  String sql = "SELECT raavare_navn FROM raavare WHERE raavare_id =
  "+id;
98                  stmt = conn.createStatement();
99                  ResultSet rs = stmt.executeQuery(sql);
100
101                 if (rs.next()) {
102                     return rs.getString("raavare_navn");
103                 } else {
104                     return "ID findes ikke!";
105                 }
106             } catch(SQLException e) {
107                 return "SQL Fejl";
108             }
109         }
110
111         public ArrayList<ReceptKomponentDTO> getRaavarerInRecept(int id) {
112             try {
113                 ArrayList<ReceptKomponentDTO> receptkomponenter = new
  ArrayList<ReceptKomponentDTO>();
114
115                 String sql = "SELECT * FROM receptkomponent WHERE recept_id = " +
  id;
116                 stmt = conn.createStatement();
117                 ResultSet rs = stmt.executeQuery(sql);
118
119                 while (rs.next())
120                 {
```

```
121                     receptkomponenter.add(new
    ReceptKomponentDTO(rs.getInt("recept_id"), rs.getInt("raavare_id"),
122                             rs.getDouble("nom_netto"), rs.getDouble("tolerance")));
123                 }
124 
125                 return receptkomponenter;
126 
127             } catch (SQLException e) {
128                 return null;
129             }
130         }
131 
132         public String getOprNameFromID(String id) {
133             try {
134                 String sql = "SELECT opr_navn FROM brugere WHERE opr_id = " + id;
135                 stmt = conn.createStatement();
136                 ResultSet rs = stmt.executeQuery(sql);
137 
138                 if (rs.next()) {
139                     return rs.getString("opr_navn");
140                 } else {
141                     return "ID findes ikke!";
142                 }
143 
144             } catch (SQLException e) {
145                 return "SQL fejl";
146             }
147         }
148 
149         public String getReceptIDFromPBID(String id) {
150             try {
151                 String sql = "SELECT recept_id FROM produktbatch WHERE pb_id = " +
    id;
152                 stmt = conn.createStatement();
153                 ResultSet rs = stmt.executeQuery(sql);
154 
155                 if (rs.next()) {
156                     return rs.getString("recept_id");
157                 } else {
158                     return "ID findes ikke!";
159                 }
160             } catch(SQLException e) {
161                 return "SQL Fejl";
162             }
163         }
164     }
165 
166 
```

```java
package ASE;

import java.io.*;

public class MainASE {

    static String response = "";
    static boolean RM20_status = false;

    static BufferedReader inputServer;
    static DataOutputStream outToServer;

    static boolean realScale = true;

    public static void main(String[] args) throws InterruptedException {
        DAL dal = new DAL();

        String opr_id, opr_name = "";
        String recept_navn, recept_id;
        String produktbatch_id;
        String tarabeholder_vaegt, afvejet_vaegt;
        String raavare_navn, raavare_amount, raavare_tolerance, raavarebatch_id;
        ArrayList<ReceptKomponentDTO> receptkomponenter = new ArrayList<ReceptKomponentDTO>();

        // Først skal operatøren logge ind

        try {
            // Forbindelse til vægten oprettes
            String response;
            Socket clientSocket;
            if (realScale) clientSocket = new Socket("169.254.2.3", 8000);
            else clientSocket = new Socket("localhost", 8000);

            outToServer = new DataOutputStream(clientSocket.getOutputStream());
            inputServer = new BufferedReader(new InputStreamReader(
                    clientSocket.getInputStream()));

            // Første to linjer spises
            response = inputServer.readLine();
            System.out.println(response);
            if (!realScale) { // Der læses kun én linje hvis det er den rigtige vægt
                response = inputServer.readLine();
                System.out.println(response);
            }

            // Skriver noget i sekundære display (nødvendigt for at vores simulator fungerer korrekt)
            if (!realScale) {
                outToServer.writeBytes("P111 \"  \"" + '\n');
                inputServer.readLine();
                inputServer.readLine();
            }

            boolean loopOne = false;
            while (true) {

                // Prompt for gyldigt operatør nummer på vægt
                do {
                    writeRM20ToScale(4, "Operator ID?", "", "");
                    opr_id = readRM20FromScale();
```

```
70                      opr_name = dal.getOprNameFromID(opr_id);
71                      // Skriv P111 hvis ID ikke findes (og vent 2 sekunder så bruger
   kan nå at læse)
72                      if ("ID findes ikke!".equals(opr_name) || "SQL
   fejl".equals(opr_name)) {
73                          System.out.println("ID FANDTES IKKE!");
74                          outToServer.writeBytes("P111 \"ID findes ikke - Proev
   igen!\"" + '\n');
75                          System.out.println(inputServer.readLine());
76                          if (!realScale) inputServer.readLine();
77                          Thread.sleep(2000);
78                      }
79                      else {
80                          resetP111();
81                      }
82                  } while ("ID findes ikke!".equals(opr_name) || "SQL
   fejl".equals(opr_name));
83
84                  loop1: while (true) {
85
86                      // Prompt for om navnet er korrekt på vægt
87                      writeRM20ToScale(8, opr_name + "?(Y/N)", "Y", "");
88                      response = readRM20FromScale().toUpperCase();
89                      if (response.equals("Y")) {
90                          // Hvis respons er Y breakes ud af while loopet
91                          loopOne = true;
92                          break loop1;
93
94                      } else if (response.equals("N")) {
95                          // Hvis respons er N skal nyt operatør nummer indtastes,
96                          // break til loop1
97                          break;
98                      } else {
99                          // Hvis respons hverken er Y eller N spørges igen ved
100                         // break til loop2
101                         continue;
102                     }
103                 }
104                 if (loopOne) {
105                     break;
106                 }
107             }
108
109             System.out.println(opr_name + " is logged in as the using operator");
110
111             // 5: Operatøren indtaster produktbatch nummer.
112
113             // Prompt for gyldigt produktbatch id
114             do {
115                 writeRM20ToScale(4, "Produktbatch ID?", "", "");
116                 response = readRM20FromScale();
117                 produktbatch_id = response;
118                 recept_id = dal.getReceptIDFromPBID(produktbatch_id);
119                 recept_navn = dal.getReceptNavnFromPBID(produktbatch_id);
120                 // Skriv P111 hvis ID ikke findes (og vent 2 sekunder så bruger kan
   nå at læse)
121                 if ("ID findes ikke!".equals(recept_navn) || "SQL
   fejl".equals(recept_navn)) {
122                     outToServer.writeBytes("P111 \"ID findes ikke - Proev igen!\""
   + '\n');
123                     inputServer.readLine();
124                     if (!realScale) inputServer.readLine();
```

```
125                     Thread.sleep(2000);
126                 }
127                 else {
128                     resetP111();
129                 }
130             } while ("ID findes ikke!".equals(recept_navn) || "SQL
    Fejl".equals(recept_navn));
131
132             // 6: Vægten svarer tilbage med navn på recept der skal produceres
    (eks: saltvand med citron)
133             outToServer.writeBytes("P111 \""+recept_navn+"\"" + '\n');
134             inputServer.readLine();
135             if (!realScale) inputServer.readLine();
136             Thread.sleep(2000);
137             resetP111();
138
139             // 16: Pkt. 7 – 15 gentages indtil alle råvarer er afvejet.
140             // Hent alle råvarer i en recept
141             receptkomponenter =
    dal.getRaavarerInRecept(Integer.valueOf(recept_id));
142
143             int i = 1;
144             for (ReceptKomponentDTO rk : receptkomponenter) {
145                 // 7: Operatøren kontrollerer at vægten er ubelastet og trykker
    'ok'
146
147                 do {
148                     writeRM20ToScale(8, "Vaegt ubelastet?(OK)", "OK", "");
149                     response = readRM20FromScale().toUpperCase();
150                 } while (!"OK".equals(response));
151
152                 // 8: Systemet sætter produktbatch nummerets status til "Under
    produktion".
153                 // Gøres kun for første råvare
154                 if (i == 1) {
155                     dal.setProduktBatchStatus(Integer.valueOf(produktbatch_id), 1);
156                     outToServer.writeBytes("P111 \"PB er nu Under Produktion\"" +
    '\n');
157                     inputServer.readLine();
158                     if (!realScale) inputServer.readLine();
159                     Thread.sleep(2000);
160                     resetP111();
161                 }
162
163                 // 9: Vægten tareres
164                 outToServer.writeBytes("T" + '\n');
165                 System.out.println(inputServer.readLine());
166                 if (!realScale) inputServer.readLine();
167
168                 // 10: Vægten beder om første tara beholder. 11: Operatør placerer
    første tarabeholder og trykker 'ok'.
169
170                 // Simulerer at en masse placeres på vægten (bruges kun med
    simulator)
171                 if (!realScale) {
172                     outToServer.writeBytes("B 0.125" + '\n');
173                     System.out.println(inputServer.readLine());
174                     inputServer.readLine();
175                 }
176
177                 do {
178                     writeRM20ToScale(8, "Saet beholder paa (OK)", "OK", "");
```

```
179                     response = readRM20FromScale().toUpperCase();
180                 } while (!"OK".equals(response));
181 
182                 // 12: Vægten af tarabeholder registreres
183                 outToServer.writeBytes("S" + '\n');
184                 tarabeholder_vaegt = inputServer.readLine().split(" ")
    [7].replaceAll(",", ".");
185                 if (!realScale) inputServer.readLine();
186 
187                 System.out.println("Beholder vægt: "+tarabeholder_vaegt);
188 
189                 // 13: Vægten tareres.
190                 outToServer.writeBytes("T" + '\n');
191                 System.out.println(inputServer.readLine());
192                 if (!realScale) inputServer.readLine();
193 
194                 raavare_navn = dal.getRaavareNameFromID(rk.getRvrId());
195 
196                 // 14: Vægten beder om raavarebatch nummer på første råvare.
197                 do {
198                     writeRM20ToScale(4, "RB ID for "+raavare_navn, "", "");
199                     response = readRM20FromScale();
200                     raavarebatch_id =
    dal.getRaavarebatch(Integer.valueOf(response), rk.getRvrId());
201                     if ("ID findes ikke!".equals(raavarebatch_id) || "SQL
    fejl".equals(raavarebatch_id)) {
202                         outToServer.writeBytes("P111 \"ID indeholder ikke
    raavaren!\"" + '\n');
203                         inputServer.readLine();
204                         if (!realScale) inputServer.readLine();
205                         Thread.sleep(2000);
206                     }
207                     else {
208                         resetP111();
209                     }
210                 } while ("ID findes ikke!".equals(raavarebatch_id) || "SQL
    Fejl".equals(raavarebatch_id));
211 
212                 // 15: Operatøren afvejer op til den ønskede mængde og trykker 'ok'
213                 raavare_amount = String.valueOf(rk.getNomNetto());
214                 raavare_tolerance = String.valueOf(rk.getTolerance());
215 
216                 // Simulér masse placeret på vægt
217                 if (!realScale) {
218                     outToServer.writeBytes("B
    "+(rk.getNomNetto()+Double.valueOf(tarabeholder_vaegt))+ '\n');
219                     System.out.println(inputServer.readLine());
220                     inputServer.readLine();
221                 }
222 
223                 do {
224                     writeRM20ToScale(8, "Afvej "+raavare_amount+" kg", "OK", "");
225                     response = readRM20FromScale();
226                     outToServer.writeBytes("S" + '\n');
227                     afvejet_vaegt = inputServer.readLine().split(" ")
    [7].replaceAll(",", ".");
228                     if (!realScale) inputServer.readLine();
229                     System.out.println("Afvejet: "+afvejet_vaegt);
230                     if (!checkWeight(afvejet_vaegt, raavare_amount,
    raavare_tolerance)) {
231                         outToServer.writeBytes("P111 \"Der er afvejet for
    lidt/meget.\"" + '\n');
```

```
232                         inputServer.readLine();
233                         if (!realScale) inputServer.readLine();
234                         Thread.sleep(2000);
235                     }
236                 } while (!"OK".equals(response) || !checkWeight(afvejet_vaegt,
    raavare_amount, raavare_tolerance));
237 
238                 dal.insertProduktBatchKomp(Integer.valueOf(produktbatch_id),
    Integer.valueOf(raavarebatch_id),
239                         Double.valueOf(tarabeholder_vaegt.replaceAll(",", ".")),
    Double.valueOf(afvejet_vaegt.replaceAll(",", ".")), Integer.valueOf(opr_id));
240 
241                 // Fortæl operatør at han er færdig med denne råvare og kan starte
    ny.
242                 outToServer.writeBytes("P111 \""+raavare_navn+" er nu afvejet.\"" +
    '\n');
243                 inputServer.readLine();
244                 if (!realScale) inputServer.readLine();
245                 Thread.sleep(2000);
246 
247                 // Skriv besked hvis der er flere råvarer der skal afvejes.
248                 if (i < receptkomponenter.size()) {
249                     outToServer.writeBytes("P111 \"Fortsaet med naeste raavare.\""
    + '\n');
250                     inputServer.readLine();
251                     if (!realScale) inputServer.readLine();
252                 }
253                 Thread.sleep(2000);
254                 resetP111();
255                 i++;
256             }
257 
258             // 17: Systemet sætter produktbatch nummerets status til ”Afsluttet”.
259             dal.setProduktBatchStatus(Integer.valueOf(produktbatch_id), 2);
260             outToServer.writeBytes("P111 \"PB er nu Afsluttet\"" + '\n');
261             inputServer.readLine();
262             if (!realScale) inputServer.readLine();
263             Thread.sleep(2000);
264             resetP111();
265 
266             // 18: Det kan herefter genoptages af en ny operatør.
267             System.out.println("Goodbye");
268 
269             clientSocket.close();
270             System.exit(0);
271 
272         } catch (UnknownHostException e) {
273             // TODO Auto-generated catch block
274             e.printStackTrace();
275         } catch (IOException e) {
276             // TODO Auto-generated catch block
277             e.printStackTrace();
278         }
279 
280     }
281 
282     private static void resetP111() throws IOException {
283         outToServer.writeBytes("P111 \"  \"" + '\n');
284         inputServer.readLine();
285         if (!realScale) inputServer.readLine();
286     }
287 
```

```
288     /**
289      * Checks whether a weight is within its tolerance range (in percentage).
290      * @param afvejet_vaegt
291      * @param raavare_amount
292      * @param raavare_tolerance
293      * @return true if weight is within range and false if it is not.
294      */
295     private static boolean checkWeight(String afvejet_vaegt, String raavare_amount,
    String raavare_tolerance) {
296         if ((Double.valueOf(afvejet_vaegt) <=
    Double.valueOf(raavare_amount)+((Double.valueOf(raavare_amount)/100.0)*Double.value
    Of(raavare_tolerance))) &&
297         (Double.valueOf(afvejet_vaegt) >= Double.valueOf(raavare_amount)-
    ((Double.valueOf(raavare_amount)/100.0)*Double.valueOf(raavare_tolerance)))) return
    true;
298         return false;
299     }
300
301     /**
302      * Sends an RM20 command to the scale.
303      *
304      * @param type
305      *             - 4 = integer, 8 = alphanum
306      * @param text1
307      *             - The string to be displayed on the scale (max. 24 chars).
308      * @param text2
309      *             - text2 Text/value to be displayed as default, and to be
310      *             overwritten by user input.
311      * @param text3
312      *             - Unit (max. 7 characters).
313      */
314     private static void writeRM20ToScale(int type, String text1, String text2,
315             String text3) {
316         try {
317             System.out.println("Sender RM20 omkring \"" + text1 + "\"");
318             // outToServer.writeBytes("RM20 4 \"Operatør nummer?\" \"\" \"\" " +
319             // "\n\r");
320             outToServer.writeBytes("RM20 " + type + " \"" + text1 + "\" \""
321                     + text2 + "\" \"" + text3 + "\"\r\n");
322         } catch (IOException e) {
323             e.printStackTrace();
324         }
325     }
326
327     private static String readRM20FromScale() throws IOException {
328         while (!RM20_status) {
329             response = inputServer.readLine();
330             if (!realScale) inputServer.readLine();
331             if (response.startsWith("RM20 B")) {
332                 //              System.out.println("Command executed, user input
    follows.");
333                 RM20_status = true;
334             } else if (response.startsWith("RM20 I")) {
335                 //              System.out.println("Command understood but not
    executable at the moment.");
336                 break;
337             } else if (response.startsWith("RM20 L")) {
338                 //              System.out.println("Command understood but
    parameter wrong.");
339                 break;
340             }
341             try {
```

```
342                 Thread.sleep(200);
343             } catch (InterruptedException e) {
344                 e.printStackTrace();
345             }
346         }
347 
348         while (RM20_status) {
349             response = inputServer.readLine();
350             if (!realScale) inputServer.readLine();
351             if (response.startsWith("RM20 A")) {
352                 response = response.split(" ")[2];
353                 // Validate here if response is an integer or string?
354                 RM20_status = false;
355                 return response.replaceAll("\"", "");
356             } else if (response.startsWith("RM20 C")) {
357                 System.out.println("RM20 afbrudt på vægt.");
358                 RM20_status = false;
359             }
360             try {
361                 Thread.sleep(200);
362             } catch (InterruptedException e) {
363                 // TODO Auto-generated catch block
364                 e.printStackTrace();
365             }
366         }
367         return "RM20 afbrudt på vægt.";
368     }
369 
370 }
371 
```

Bilag 17 - Simulator kildekode

```java
package boundary;

import java.io.BufferedReader;

public class ClientConnection implements Runnable {
    IFunction func;

    private Socket client;
    private BufferedReader in = null;
    private PrintWriter out = null;

    public ClientConnection(Socket CC, IFunction func){
        this.func = func;
        this.client = CC;
        try {
            in = new BufferedReader(new InputStreamReader(client.getInputStream()));
            out = new PrintWriter(client.getOutputStream(), true);
            out.println("Forbundet til Mettler Vægt Simulator.");
            out.println("Vægtens port: "+client.getLocalPort()+". Lokal port: "+client.getPort());
        } catch(IOException e) {
            System.err.println(e);
            return;
        }
    }


    @Override
    public void run() {
        String input, response;
        while(true) {
            try {
                if (func.getRM20Answer() != "") { // svarer hvis der er indtastet et RM20 svar
                    out.println("RM20 A "+func.getRM20Answer()+"\r");
                    func.restoreDisplay();
                    func.setRM20Answer("");
                }
                else if (in.ready()){
                    input = in.readLine();
                    if (input != null) {
                        response = func.interpret(input, true);
                        out.println(response + "\r");
                    }
                }
            } catch (IOException e) {
                System.err.println(e);
            }
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
}
```

```java
package entity;

public class Entity implements IEntity {
    private double brutto = 0.000;
    private double tara;
    private boolean RM20;
    private String mainDisplay = "";
    private String secDisplay = "";
    private String text1 = "";
    private String text2 = "";
    private String text3 = "";
    private String RM20Answer = "";
    private String storedSecDisplay = "";
    private String storedPrimDisplay = "";

    public Entity() {
        tara = 0.000;
        RM20 = false;

    }

    /* (non-Javadoc)
     * @see entity.IEntity#getBrutto()
     */
    @Override
    public double getBrutto() {
        return brutto;
    }


    /* (non-Javadoc)
     * @see entity.IEntity#setBrutto(double)
     */
    @Override
    public void setBrutto(double brutto) {
        this.brutto = brutto;
    }


    /* (non-Javadoc)
     * @see entity.IEntity#getTara()
     */
    @Override
    public double getTara() {
        return tara;
    }


    /* (non-Javadoc)
     * @see entity.IEntity#setTara(double)
     */
    @Override
    public void setTara(double tara) {
        this.tara = tara;
    }

    @Override
    public boolean getRM20() {
        return RM20;
    }

    @Override
```

```java
    public void setRM20(boolean RM20) {
        this.RM20 = RM20;
    }

    @Override
    public void setText(String text) {
        mainDisplay = text;

    }

    @Override
    public String getText() {
        return mainDisplay;
    }

    public String getSecDisplay() {
        return secDisplay;
    }

    public void setSecDisplay(String secDisplay) {
        this.secDisplay = secDisplay;
    }

    @Override
    public String gettext1() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public String settext1(String text) {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public String gettext2() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public String settext2(String text) {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public String gettext3() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public String settext3(String text) {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public String getRM20Answer() {
        return RM20Answer;
```

```
125     }
126
127     @Override
128     public void setRM20Answer(String text) {
129         RM20Answer = text;
130     }
131
132     @Override
133     public void storeDisplay() {
134         // TODO Auto-generated method stub
135         storedPrimDisplay = mainDisplay;
136         storedSecDisplay = secDisplay;
137     }
138
139     @Override
140     public void restoreDisplay() {
141         mainDisplay = storedPrimDisplay;
142         secDisplay = storedSecDisplay;
143         // TODO Auto-generated method stub
144     }
145
146 }
147
```

```java
package function;

import boundary.Menu;

public class Function implements IFunction {
    Menu menu;
    IEntity data;

    public Function(IEntity data) {
        this.data = data;
    }

    public void setBoundary(Menu menu) {
        this.menu = menu;
    }

    @Override
    public String interpret(String input, boolean extCmd) {
        if (input.equals("T")){ // Tar□r v□gt
            tareWeight();
            return "T S      " + String.format("%.3f",getTara()) + " kg";
        }
        else if (input.startsWith("B ")) {
            try {
                if(!input.contains(".")) {
                    input += ".";
                }
                input += "0000";
                changeWeight(Double.parseDouble(input.substring(2,7)));
                return "DB";
            } catch(NumberFormatException e) {
                return "Input fejl, prøv igen.\nIndtast kommando:";
            }
        }
        if (extCmd) { // Kommandoer kan kun bruges af en ekstern klient
            if(getRM20()) {
                return "RM20 I";
            }
            else if (input.equals("S")) {
                if(getWeight() < 0) {
                    return "S S      " + String.format("%.3f",getWeight()) + " kg";
                }
                return "S S      " + String.format("%.3f",getWeight()) + " kg";

            }
            else if (input.equals("T")){ // Tar□r v□gt
                tareWeight();
                return "T S";
            } else if (input.startsWith("D ")) {
                try{
                    if (input.split(" ")[1].split("\"")[1].length() <= 7) {
                        displayMsg(input.substring(3, input.lastIndexOf("\"")));
                        return "D A";
                    }
                }catch (IndexOutOfBoundsException e){
                    return  "ES";
                }
                return "ES - Message too long (max. 7 chars)";

            } else if (input.equals("DW")){
                displayMsg(" ");
                return "DW A";
```

```
64
65              }
66
67              else if(input.startsWith("P111 ")){
68                  try {
69                      if (input.split(" \"")[1].length() < 30) {
70                          displaySecMsg(input.substring(6, input.lastIndexOf("\"")));
71                          return "P111 A";
72                      }
73                  } catch (IndexOutOfBoundsException e) {
74                      return "ES";
75                  }
76                  return "ES - Message too long (max. 30 chars)";
77              }
78              else if (input.startsWith("RM20 ")){
79                  engageRM20(true);
80                  try {
81                      storeDisplay();
82                      String split[] = input.split(" ");
83                      if(split[1].equals("4") || split[1].equals("8")) {
84                          split = input.split("\"");
85                          if (split[1].length() < 24)
86                              data.setSecDisplay(split[1]);
87                          else {
88                              restoreDisplay();
89                              engageRM20(false);
90                              return "ES - Message too long (max. 24 chars)";
91                          }
92                      }
93                      else {
94                          restoreDisplay();
95                          engageRM20(false);
96                          data.setSecDisplay("");
97                          return "RM20 L";
98                      }
99
100                 } catch(IndexOutOfBoundsException e) {
101                     restoreDisplay();
102                     engageRM20(false);
103                     data.setSecDisplay("");
104                     return "RM20 L";
105                 }
106                 return "RM20 B";
107
108             }
109         }
110         else { // Kun tilgængelig via vægtens lokale konsol
111             if (input.equals("Q")){
112                 System.out.println("Systemet lukker ned.");
113                 System.exit(1);
114             }
115         }
116         if (extCmd)
117             return "ES";
118         else
119             return "ES\nIndtast kommando:";
120     }
121
122     @Override
123     public String getWeightString() {
124         return Double.toString(data.getBrutto()-data.getTara());
125     }
```

```
126
127     @Override
128     public void tareWeight() {
129         data.setTara(data.getBrutto());
130     }
131
132     @Override
133     public void displayMsg(String msg) {
134         data.setText(msg);
135     }
136
137     @Override
138     public void displayWeight() {
139     }
140
141     @Override
142     public void displaySecMsg(String msg) {
143         data.setSecDisplay(msg);
144     }
145
146     @Override
147     public void changeWeight(double weight) {
148         if(weight > 6) {
149             System.out.println("Brutto må ikke overstige 6 kg");
150         } else {
151             data.setBrutto(weight);
152         }
153     }
154
155
156     @Override
157     public void quit() {
158         System.out.println("Systemet lukker");
159         System.exit(1);
160         //luk alle connections
161     }
162
163     @Override
164     public void zeroWeight() {
165         data.setBrutto(0.0);
166         data.setTara(0.0);
167     }
168
169     @Override
170     public Double getWeight() {
171         return data.getBrutto()-data.getTara();
172     }
173
174     @Override
175     public boolean getRM20() {
176         return data.getRM20();
177     }
178
179     @Override
180     public String getText() {
181         return data.getText();
182     }
183
184     @Override
185     public String getSecText() {
186         // TODO Auto-generated method stub
187         return data.getSecDisplay();
```

```
188     }
189
190     @Override
191     public double getBrutto(){
192         return data.getBrutto();
193     }
194
195     @Override
196     public double convert(String input){
197         return Double.parseDouble(input) + 1;
198     }
199
200     @Override
201     public double getTara() {
202         return data.getTara();
203     }
204
205     @Override
206     public void engageRM20(boolean state) {
207         data.setRM20(state);
208     }
209
210     @Override
211     public String getRM20Answer() {
212         return data.getRM20Answer();
213     }
214
215     @Override
216     public void setRM20Answer(String text) {
217         data.setRM20Answer(text);
218     }
219
220     @Override
221     public void storeDisplay() {
222         data.storeDisplay();
223         // TODO Auto-generated method stub
224     }
225
226     @Override
227     public void restoreDisplay() {
228         data.restoreDisplay();
229     }
230 }
231
```

```java
package entity;

public interface IEntity {

    public abstract double getBrutto();

    public abstract void setBrutto(double brutto);

    public abstract double getTara();

    public abstract void setTara(double tara);

    public abstract void setRM20(boolean RM20);

    public abstract boolean getRM20();

    public abstract String getRM20Answer();
    public abstract void setRM20Answer(String text);

    public abstract void setText(String text);

    public abstract String getText();

    public abstract String getSecDisplay();

    public abstract void setSecDisplay(String secDisplay);

    public abstract String gettext1();
    public abstract String settext1(String text);
    public abstract String gettext2();
    public abstract String settext2(String text);
    public abstract String gettext3();
    public abstract String settext3(String text);

    public abstract void storeDisplay();
    public abstract void restoreDisplay();
}
```

```java
package function;

import boundary.Menu;

public interface IFunction {
    void setBoundary(Menu menu);
    String interpret(String input, boolean b);
    String getWeightString();
    Double getWeight();
    boolean getRM20();
    void engageRM20(boolean state);
    void tareWeight();
    void displayMsg(String msg);
    void displayWeight();
    void displaySecMsg(String msg);
    void changeWeight(double weight);
    void zeroWeight();
    void quit();
    String getText();
    String getSecText();
    double getBrutto();
    double convert(String input);
    public double getTara();
    public String getRM20Answer();
    public void setRM20Answer(String text);
    public void storeDisplay();
    public void restoreDisplay();
}

```

```java
package boundary;

import java.util.Scanner;

public class InputThread extends Thread {
    IFunction func;

    public InputThread(IFunction func) {
        this.func = func;
    }

    public void run(){
        Scanner sc = new Scanner(System.in);
        String input;
        while(true) {

            try {
                if (func.getRM20()) System.out.println(func.getSecText()+":");
                input = sc.nextLine();
                if (func.getRM20()) {
                    func.engageRM20(false);
                    func.setRM20Answer(input);
                }
                else {
                    System.out.println(func.interpret(input, false));
                }
                Thread.sleep(120);
            } catch (java.util.InputMismatchException e) {
                System.out.println("Indtast korrekt input.");
                sc.nextLine();
            } catch (InterruptedException e) {
                System.out.println("God historie 3");
            }
        }
    }
}

```

```java
package boundary;

import function.IFunction;

public class Listener extends Thread {
    IFunction func;
    private static int port;

    public Listener(IFunction func, int port) {
        this.func = func;
        this.port = port;
    }

    @Override
    public void run() {
        ServerSocket server  = null;
        try {
            server = new ServerSocket(port);
        } catch (IOException e) {
            System.err.println("Kunne ikke lytte på port " + port + ". Husk at terminere tidligere program eller angiv anden lytte port.");
            System.err.println(e);
            System.exit(1);
        }
        Socket client = null;
        while(true) {
            try {
                client = server.accept();
            } catch (IOException ie) {
                System.out.print("Fangede IO fejl: " + ie.getMessage());
                System.exit(1);
            }
            // Connection started
            Thread CC = new Thread(new ClientConnection(client,func));
            CC.start();
        }
    }
}

```

```java
import boundary.Menu;


public class Main {

    public static void main(String[] args) {

        int listenPort = 8000;
        if (args.length > 0) {
            try {
                listenPort = Integer.parseInt(args[0]);
            } catch (NumberFormatException e) {
                System.err.println("Fejl i argument. Argument angiver portnummeret der skal lyttes på og skal være en integer.");
                System.exit(1);
            }
        }

        IEntity data = new Entity();
        IFunction func = new Function(data);
        Listener list = new Listener(func, listenPort);
        Menu menu = new Menu(func);
        func.setBoundary(menu);


        list.start();
        menu.run();
    }

}

```

```java
package boundary;

import function.IFunction;

public class Menu extends Thread  {

    String mainDisplay = " ";
    String secDisplay = "";
    IFunction func;
    Double brutto = 0.000;
    Double tara = 0.000;
    Double displayed = 0.000;
    Boolean displayrm20 = false;

    public Menu(IFunction func) {
        this.func = func;
    }

    @Override
    public void run() {
        displayrm20 = false;
        InputThread input = new InputThread(func);
        try {
            input.start();

            System.out.println("============================");
            System.out.println("   Mettler Vægt Simulator  ");
            System.out.println("============================");
            System.out.printf("Primær display:     %.3f kg \n", displayed);
            System.out.println("Sekundær display: "+ secDisplay);
            System.out.print("Indtast kommando: ");

            while (true) {

                if(!func.getRM20()) {
                    if(!func.getText().equals("") &&
  !func.getText().equals(mainDisplay) || (!func.getSecText().equals("") &&
  !func.getSecText().equals(secDisplay))) {
                        displayrm20 = false;

                        System.out.println("\n\n\n");
                        System.out.println("============================");
                        System.out.println("   Mettler Vægt Simulator     ");
                        System.out.println("============================");

                        if(!func.getText().equals(" ") &&
  (!func.getText().equals("") && !func.getText().equals(mainDisplay))) {
                            mainDisplay = func.getText();
                            System.out.println("Primær display: " + mainDisplay);
                        } else if (func.getText().equals(" ") &&
  !mainDisplay.equals(" ")) {
                            mainDisplay = func.getText();
                            displayed = func.getBrutto()-func.getTara();
                            System.out.printf("Primær display:     %.3f kg \n",
  displayed);
                        } else if(func.getText().equals(" ")) {
                            mainDisplay = "";
                            func.displayMsg("");
                            displayed = func.getBrutto()-func.getTara();
                            System.out.printf("Primær display:     %.3f kg \n",
  displayed);

```

```
57                          } else if((!func.getSecText().equals("") &&
  !func.getSecText().equals(secDisplay)) && !mainDisplay.equals("") &&
  mainDisplay.equals(" ")) {
58                              mainDisplay = func.getText();
59                              System.out.println("Primær display: " + mainDisplay);
60                          } else {
61                              System.out.println("Primær display: " + mainDisplay);
62                          }
63
64                          secDisplay = func.getSecText();
65                          //Else brutto
66                          mainDisplay = func.getText();
67                          System.out.println("Sekundær display: " + secDisplay);
68                          System.out.print("Indtast kommando: ");
69
70                      } else if((func.getBrutto()-func.getTara() != displayed &&
  func.getText().equals("")) || !func.getText().equals(" ") &&
  func.getText().equals("") && (func.getBrutto() != brutto ||
  !func.getText().equals("") || func.getTara() != tara ||
  (!func.getSecText().equals("") && !func.getSecText().equals(secDisplay)))) {
71                          //Dette vises ved ændring i brutto
72                          mainDisplay = func.getText();
73                          secDisplay = func.getSecText();
74
75                          brutto = func.getBrutto();
76                          tara = func.getTara();
77                          displayrm20 = false;
78                          System.out.println("\n\n\n");
79                          System.out.println("============================");
80                          System.out.println("   Mettler Vægt Simulator  ");
81                          System.out.println("============================");
82                          displayed = func.getBrutto()-func.getTara();
83                          System.out.printf("Primær display:     %.3f kg \n",
  displayed);
84                          System.out.println("Sekundær display: "+ secDisplay);
85                          System.out.print("Indtast kommando: ");
86                      }
87                  } else {
88                      if(displayrm20 == false) {
89                          mainDisplay = func.getText();
90                          secDisplay = func.getSecText();
91                          System.out.println("\n\n\n");
92                          System.out.println("============================");
93                          System.out.println("   Mettler Vægt Simulator  ");
94                          System.out.println("============================");
95                          System.out.printf("Primær display:   \n");
96                          System.out.printf("Sekundær display: %s\n",
  func.getSecText());
97                          System.out.print("Indtast kommando: ");
98                          displayrm20 = true;
99                      }
100                 }
101                 Thread.sleep(100);
102             }
103         } catch (InterruptedException e) {
104             System.out.println("God historie 1");
105         }
106     }
107 }
108
```