Star Mobil

Elva Hartmannsdóttir

Kongens Lyngby 2007
IMM–B.Eng.-2007-19

Resumé

I denne projekt har jeg undersøgt om det er relevant at udvikle en mobil web applikation af Star Control® og om den vil være brugbare produkt for den som arbejder på et renseanlæg. Star Control® overvågner renseprocessen på et renseanlæg. Web applikationen skal kunne give brugeren status over processen. Dette projekt har jeg lavet i samarbejde med Krüger A/S i Søborg.

Applikationen kan besøges på adressen http://grock.star2.dk:28080/StarMobile

Brugernavn: star

Password: starUser.

Forord

Denne rapport(projekt) er udarbejdet ved Informatik og Matematisk Modellering(IMM) på Danmarks Tekniske Universitet og i samarbejde med Krüger A/S. projektet er den sidste del i at færdiggøre min IT diplomingeniør uddannelse.

Denne rapport beskriver hvordan jeg har lavet en web applikation for mobil telefon af Star Control. Hovedopgaven var at lave en web applikation for en mobile telefon som vil være brugbar ude på et renseanlæg og kunne vise brugeren proces statusen på renseanlægget.

Denne rapport består af resumé, beskrivelse af løsning og bilag skrevet i foråret 2007.

Gladsaxe, april 2007
Elva Hartmannsdóttir

Anerkendelse

Jeg vil gerne takke alle dem som har hjulpet og støttet mig igennem hele projektforløbet. Særlig alle dem som arbejder i SCRUM temaet, Star gruppen i Krüger A/S og min vejleder fra DTU Niels-Ole Christensen for gode råd.

Brugte forkortelser i rapporten

API	Applikation programming interface
ATS	Aeration Tank Settling
BioP	Biologisk fosfor-fjernelse
DD	Deployment Descriptor
DOM	Document object model
GUI	Graphical User Interface
HTML	HyperText Markup Language
JAAS	Java Authentication & Authorization –service
JAR	Java Archive
JDBC	Java Database Connectivity
JNDI	Java Naming and Directory Interface
JSP	JavaServer Page
J2SE5	Java Platform 2, Standard Edition 5
MVC	Model-View-Controller arkitektur
SCADA	Supervisory Control And Data Acquisition
SRO	Styring, Regulering og Overvågning
SQL	Structured Query Language
TDD	Test driven development
URL	Uniform Resource Locator
UTC	Coordinated Universal Time
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language
WAR	Web Archive

Indholdsfortegnelse

1 Problemstilling

I denne her kapitel vil jeg skive min problem stilling for projektet og så bagefter vil jeg prøve at forklare processen på et renseanlæg

1.1 Mobil løsning for STAR control®

Krüger A/S har et software produkt STAR control som øger kapacitet og sparer penge via forbedret udnyttelse af energi og kemikalier.

STAR control er et alternativ til udbygning af renseanlæg og renovering af afløbssystemet. STAR control er især en fordelagtig løsning, hvis renseanlægget har problemer med at overholde udløbskravene, renseanlægget er for højt eller for lavt belastet eller har svigende belastning f.eks. fra industri eller sommerhusområder.

STAR control er et unikt overordnet styringssystem, der i samarbejde med det almindelige SRO-system tilpasser styringen af renseanlægget efter de faktiske og forudsigelige forhold. Det sker ved hjælp af indbyggede procestekniske erfaringer og online målinger af: ilt, flow, ammonium, nitrat og fosfat. STAR beregner den optimal styring af anlægget baseret på den aktuelle spildevandsbelastning, den aktuelle biologiske aktivitet og tidligere proceserfaringer.

Figur 1

STAR er brugervenlig, intuitiv, webbaseeret online styring af renseanlægget. STAR har en alarmhåndtering.

STAR brugergrænsefladen har nogle undergrupper, de er.

- **Proces** viser detaljer om de enkelte styringsmoduler i STAR. Modulerne er: Fase- og iltsætpunkt styring, returslam styring, metal styring, ATS styring, BioP styring, dataopsamling i afløbssystemet og individuelt tilpassede moduler.
- **Rapporter** vises som grafer, tekst og logo, af opsamlede og beregnede data som gemmes hvert 2 minut.
- **Journal** over historiske og nuværende styringsstrategier. Journal er automatisk logbog til registrering af alle ændringer i STAR styringen, bruges også til dialog mellem flere bruger.
- **Indstillinger** gøre det muligt for at ændre på styringerne og historisk følge ændringerne, men det kræver at man har login adgang i systemet.

Mit eksamensprojekt vil handle om at programmere en mobil løsning for STAR. Det betyder at de relevante oplysninger, fra proces og indstillinger, som er beskrevet oven på vil være tilgængelig på en mobiltelefon.

Løsning på projektet vil være at lave en web-server hvor mobiltelefonen kan hente en hjemmeside, som kræver at brugeren har adgang til systemet. Web-serveren skal kunne hente data fra en database som vises på hjemmesiden, data er for enkelte renseanlæg. Hjemmesiden skal kunne vise de relevante data for brugeren og de skal være i det rigtige format for en mobiltelefon, dvs. at det som vises på telefonen skal være læselig og at brugeren nemt kan arbejde med dem. Web-serveren skal også kunne modtage data fra hjemmesiden til at ændre på styringerne, men det kræver at brugeren har rettighed til det.

Til at løse dette problem vil jeg starte med at bruge en mobil simulator til at vise hjemmesiden, fordi den er hurtigere at få en forbindelse til en web-server og det bliver meget nemmer og hurtigere at bruge tastaturen på computeren end på en mobiltelefon. Hvis jeg har tid til sidst vil jeg gerne få løsningen til at køre på en mobiltelefon.

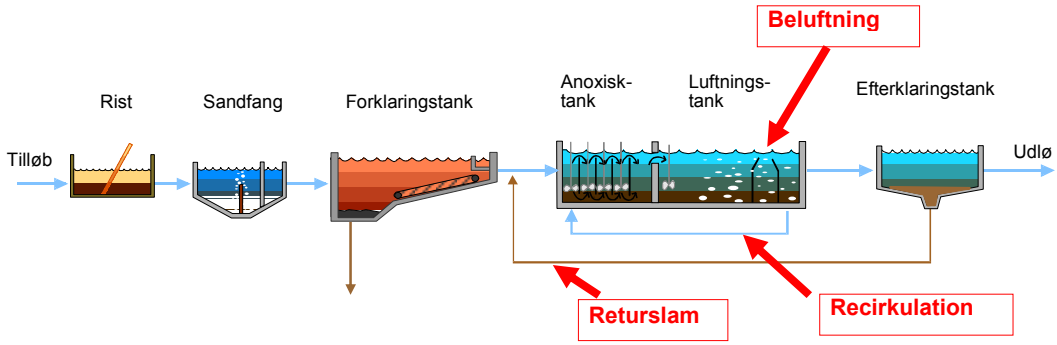
1.2 Hvad er et renseanlæg

Renseanlæg er hvor alt spildevand fra husholdninger og industrien er behandlet, dvs. alle belastende stoffer for miljøet er fjernet fra vandet. Spildevandet

er derfor blanding af mange forskellige affaldsprodukter som kan indeholde bl.a. sand fra vejene, fedt og andet affald fra køkkener, afføring og urin fra toiletter, vaskemidler fra vaskemaskiner, industrispildevand o.fl.

Spildevandet strømmer til renseanlægget gennem byens kloakledninger. Der bliver det renset til rent vand. Når de forskellige stoffer skal fjernes fra spildevandet sker det i nogle trin indtil vandet bliver så rent at det kan løbe ud i vandmiljøet uden at belaste miljøet. [litt. 4]

1.2.1 Processen



Figur 2

1.2.2 Forbehandling

Det første trin i behandlingen er når spildevandet passerer mekaniske riste. Ristene holder alle større urenheder som klud, papir, træstykker m.m. tilbage. Ristene renses automatisk, og riststoffet bliver forbrændt i en slamforbrændingsovn.

Fra ristene ledes spildevandet til det beluftede sand- og fedtfang, hvor sand og fedt udskilles, hvor sandet bundfældes og fedtudskiles på vandoverfladen. Det oppumpede sand vaskes for urenheder i sandvasker, hvorefter det bliver transporteret til en slamforbrændingsovn. Fedtet pumpes til rådnetankene, hvor det nedbrydes til biogas.

Fra sand og fedtfanget ledes spildevandet til forklaringstank. Her bundfældes indholdet af tungere slampartikler, således at der opnås en 40-50 % reduktion af det organiske stof i spildevandet. Det bundfældede slam pumpes til koncentreringstanken og derfra videre til rådnetankene. Det mekaniske rensede vand ledes til det biologiske behandlingstrin.

1.2.3 Biologisk vandbehandling

Den biologiske vandbehandling renser spildevandet for kvælstof og fosfor efter BIO-DENIPHO processen, det trin er et aktiv slamanlæg.

Biologiske behandlingen består af en Bio-P tank, en luftnings tank og en efterklaringstank. Til biologiske spildevandsrensning kræves ilt. Ilten piskes ind i spildevandet af overfladebeluftere. Omrører anvendes for at sikre at vand og slam holdes opblandet.

Biologisk rensning af spildevand er basseret på de mange forskellige mikroorganismer, der naturligt forekommer i spildevand. Mikroorganismerne omsætter og nedbryder de forurenende stoffer i spildevandet. Ved styring af processerne på renseanlægget sikres optimale levevilkår for de organismer, som har størst betydning for rensning af spildevandet. For at fjerne fosfor og kvælstoffet drejer det sig om følgende tre typer mikroorganismer:

De nitrificerende(NH_4+O_2 > NO_3) bakterier *omsætter ammoniak kvælstof, som findes i tilledt spildevandt, til nitrat-kvælstof. Denne proces sker under iltrige(aerobe) forhold. De nitrificerende bakterier vokser relativt langsomt og skal have god tid til denne process.*

De denitrificerende(NO_3 + BOD > N_2) bakterier *omsætter den dannede nitrat til luftformig kvælstof. Bakterierne omsætter organisk stof ved forbrug af den bunde ilt, der indgår i den nitrat, som de nitrificerende bakterier har dannet(anoxisk proces). Herved frigøres ren kvælstof, som forsvinder op i atomsfæren, der i forvejen består af 80% kvælstof.*

De fosforakkumulerende bakterier *er en særlig bakterietype, som er i stand til at optage ekstra store fosformængder i cellerne. Fosforen anvendes som energikilde under opformeringen, som sker under iltfrie(anaerobe) forhold ved omsætning af organisk stof fra det tilledte spildevand. Fosfor frigives samtidig fra cellerne. Under iltrige forhold(nitrifikationstrinnet) optager de fosforakkumulerende bakterier igen store fosformængder i cellerne. Fosforfjernelse sker herefter med overskudsslammet.*

[litt. 3 Renseanlægget Damhusåen]

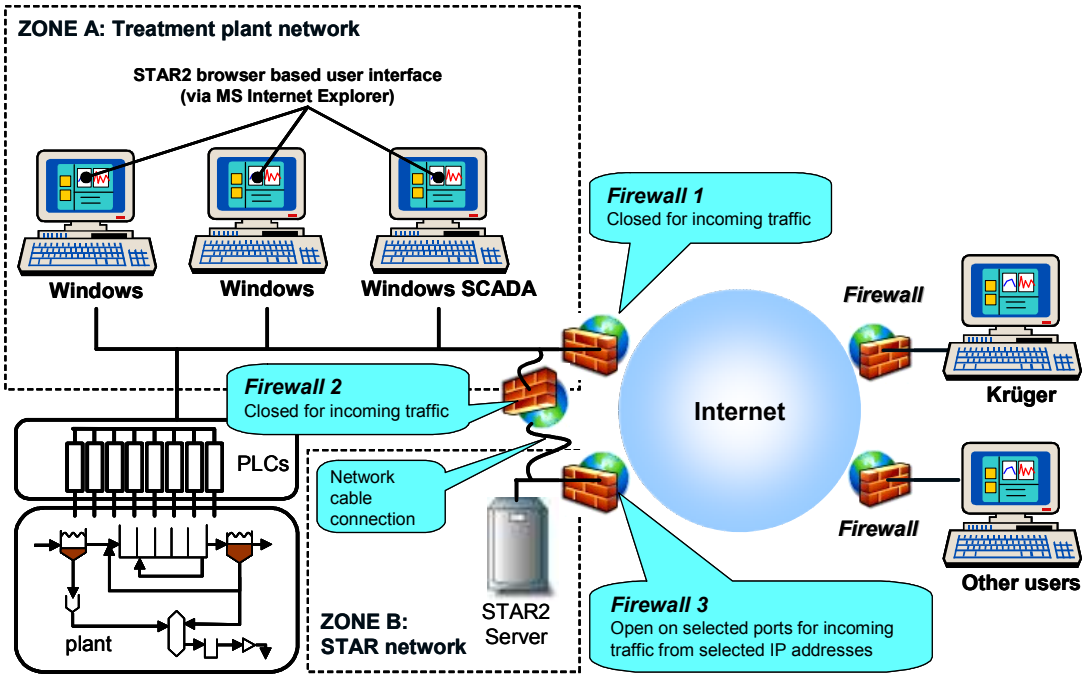
1.2.4 Efterbehandling

Fra biologisk tanken ledes vandet til en efterklaringstank, hvor det rensede vand adskilles fra det biologiske slam. Slammet bundfælles og føres tilbage til en biologisk tank. En mindre del af slammet tages ud som overskudsslam.

1.2.5 Slambehandling

Ved renseprocesserne produceres slam, i forklaringstanken som primærslam og i luftningstanken som overskudslam.
Primærslammet opkoncentreres og pumpes til rådnetanken. Her nedbrydes slammet under dannelse af biogas(methan og kuldioxid). Det biologiske overskudslam føres til forafvandere. Herefter blandes det med udrådnet primærslam. Efter det bliver slamet transporteret til en slamforbændingsovn, hvor det tørres og brændes, varmen udnyttes til bl.a. fjernvarme. [litt. 3]

1.2.6 Hvilke moduler målers og styres



Figur 3

Star online system holder konstant overvågning af ilt, ammonium, nitrat og fosfor i den biologiske proces behandling af spildevandet. Målerne er placeret på de procesmæssige rigtige steder i tankerne og data fra målerne er flyttet(transporteret) til hoved kontrolsystemet. Star kan tillægges til den eksisterende SCADA/SRO system,

og har indbygget viden til at analysere indkommet data og forslå ændringer på anlægges styrings proces.

Længden på nitrifikation og denitrifikation fase er bestemt efter de konkrete mængder af nitrat og ammonium og kører ikke i fastsæt tid. Resultat af det er at mængden af nitrat og ammonium betydeligt reduceret.

Med Star opnår man som regel betydelige energibesparelse og ofte reduktion af kemikalieforbruget, en forbedret afløbskvalitet og (forbedre) udnyttelse af værende anlægskapacitet. Star er med til at udskyde eller helt undgå udvidelser af anlægget. Hvis Star er en del af anlægges design, fra starten kan tanke volumen blive 15 % - 20 % procent mindre, og der med at gøre anlægget billigere. [litt. 5]

2 Analyse af systemet

I denne kapitel vil jeg prøve at definere de krav som systemet skal opfylde. Jeg har valgt at dele system kravene op i to grupper. Den ene gruppe er kravene som en bruger af Star control har for systemet og den anden gruppe med min egen system krav.

Til at finde ud af hvordan systemet(brugergrænsefladen) skulle opbygges startede jeg med at lave små test med statiske hjemmesider, som viste graf og nogle af de komponenter som bruges i Star control. Det gjorde jeg fordi jeg synes det var svært at finde ud af hvad telefonen kunne vise. Jeg tror også at det vil være letter for brugeren, som jeg vil snakke med, at danne sig et billede af de informationer som en mobil telefon kan vise.

2.1 Brugerens krav for applikationen

Til at finde kravene for systemet snakkede jeg med en bruger til at finde ud af, hvad vil være relevant at se på telefonen. Her er de krave som brugeren synes at applikationen skulle opfylde:

- få groft overblik over status på anlægget. Det første som man ser når man åbner Star2 er nogle grafer og en status over anlægget. Den status synes han at det skulle være det første som brugeren vil se når han er blevet logget ind på siden.
- vise det som ligger under fanebladet **Proces** på Star control. Der kan man se forskellige målinger i processen og status over hvad det sker nu i den enkelte proces. Mobiltelefonen skal kunne vise grafer og med en mulighed for at se status på den enkelte proces modul.
- se statusen på måler, med den mulighed at slå en måler fra. Det vil være en god mulighed for brugeren at kunne det, fordi mange alarmer som kommer fra systemet er på grund af en måler som ikke virker. Så vil det være rart at kunne slå måleren fra med det samme hvis brugeren ikke kan komme med det samme ind på Star control via sin computer. Det vil også være en gode mulighed for dem som arbejder ud på anlægget at kunne slå en måler fra på stedet hvis de skal reparere den ellers skulle de gå ind i kontrolrummet og åbne Star control

til at slå måleren fra. Det gør de ikke i dag, simpelthen fordi det tager alt for lang tid.

- kunne se journal og skrive nogle beskeder ind, fordi så kan medarbejder som arbejder på anlægget kunne følge med hvad de andre har lavet og hvad styringen har ændret.
- se alarm status på side, hvor brugeren vil have mulighed for at kvittere for en alarm, brugeren kunne bede om at få påmindelse om alarmen efter kort tid igen, kunne se hvilken type alarm det er, og om det er system eller proces alarm. Denne feature har Star control ikke endnu, men vil snart implementeret.

Efter snakket med brugeren gik jeg så i gang med at udvikle systemet. Da jeg havde udviklet hoved funktionen i applikationen snakkede jeg så med en anden bruger som har været en driftsleder på et anlæg hvor Star kører til at få kommentar fra en som er vant til at brug Star. Han synes det var en gode ide at få lavet en mobile løsning for de mennesker som arbejder på anlægget og det vil kunne hjælpe dem meget. Det vigtigste som han synes at man skulle kunne se på en mobil telefon var proces status på anlægget lige nu og hvordan processen har udviklet sig de sidste par timer. Det vil være meget nemmere for en driftsleder som måske sider på møde, ude at rejse eller er ud at arbejde(ikke ved computeren). Driftslederen kan se statusen på anlægget nu og så kan han f.eks. vejlede dem som er på anlægget hvordan de skal ændre nogle sætpunkter. Det er nemlig tit sådan at de små anlæg kun har en som kender alt om processen, men hvis han skal på ferie eller nogle andre steder så skal han lære en ny op til at passe anlægget. Derfor vil det være smart for en driftsleder at kunne kigge på sin telefon til at vejlede den som sider på anlægget. Vi snakkede også om at det ville være noget som vil rent faktisk bruges, i dag har alle en mobil telefon med, man behøver ikke slave med noget ekstra(f.eks. en bærbare computer) og det vil ikke tage så lang tid til at se de nødvendige informationer.

2.2 Krav specifikation

- Systemet skal være så generelt som muligt og i første omgang vise de samme informationer for alle renseanlæg, ligegyldig om modulen bruges på anlægget.
- Brugergrænsefladen skal vises som en web side, som brugeren kan se i sin browser på mobiltelefonen.
- Serveren skal kunne identificere brugerne til at give dem adgang til hjemmesiderne, så brugeren kan følge med i rense processen.
- Mobil klienten skal kunne vise de relevante oplysninger for brugeren. De skal være tilgængelige via en hjemmeside og på forskellig formater, f.eks. som tekst, graf og logo(moduler).
- Graferne bliver dannet når det kommer forespørgsel om siden. Hvis brugeren har lyst til at følge med renseanlægs processen på mobiltelefonen så skal han have mulighed for at log ind på systemet, identificere sig selv for systemet, uden at bruge tastaturen på telefonen alt for meget.
- Data som skal præsentere i mobilen bliver hentet fra en database.
- Når systemet er kommet op at køre så vil det være udviklet til at være mere specifik for det renseanlæg det kører på, fordi renseanlæggene har brug for forskellige moduler, dvs. at hjemmesiden vil kunne vise de moduler som dette renseanlæg bruger.
- Det skal være muligt for brugeren at indtaste oplysninger fra klients side, men så skal brugeren have rettighed til det.

2.3 Uden for projektsgrænser

Der er nogle ting som jeg ikke har beskæftiget mig med, men det har været brugt i projektet:

- Serveren, en web server som allerede var opsat i Krüger
- Database, opsætning og design, den ligger på et kørende anlæg
- Transform sikkerhed af data

De ting som er skrevet oven på har jeg ikke beskæftige mig med, fordi tiden som projektet skulle løses på var for kort.

2.4 Valg af løsning

Til at løse de her krav som er beskrevet her oven på har jeg tænkt mig at løse dem i korte perioder. Først har jeg tænkt mig at få en enkelt hjemmeside til at vises på en mobiletelefon. Derefter vil jeg arbejde på at få de små features som systemet skal opfylde til at virke. Se tidsplan i bilag A.

Efter hver periode vil jeg så vurdere om hvad skal løses næst og om det er kommet nogle nye krav som skal løses før end dem som jeg havde planlagt at løse i næste periode.

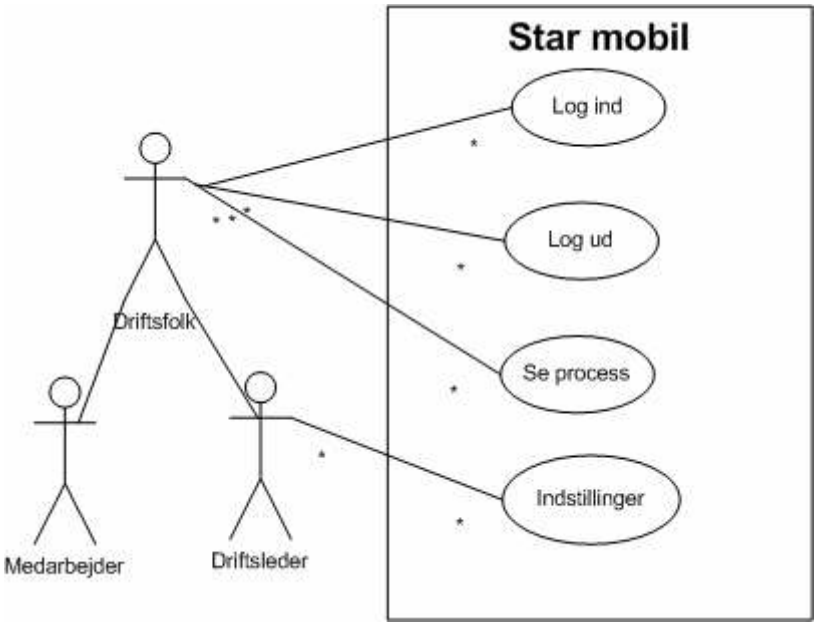
I denne projekt vil jeg brug "test driven development"(TDD) ved udviklingen af systemet så meget som muligt, dvs. at jeg vil først skrive testen som vil indeholde de krav som featuren skal opfylde, bagefter vil jeg skrive koden for featuren. Featuren er færdig når den består alle testene.

Jeg vil udvikle systemet i Java ved hjælp af JUnit, så det bliver nemmere at test hver enkelt feature. Hjemmesiderne bliver skrevet i jsp, med xhtml syntaks, fordi at web browser i en mobiltelefon er ikke lige så effektiv som en almindelig browser.

2.5 Målsætning for projektet

Min første målsætning for projektet er at udvikle et system som kører. Den næste er at udvikle systemet så at det vil være brugbart ude på renseanlæggene og opfylde de krav som brugerne har, dvs. det skal være brugervenligt og nemt at bruge. Systemet skal kunne passe sig til proces modulerne på de enkelte renseanlæg. Til sidst skal være nemt at tilføje det på nye anlæg og koden skal være let at vedligeholde.

2.6 Bruger typer af applikationen



Figur 4

Efter at have lavet krav specifikation for systemet har jeg prøvet at finde de bruger typer som vil bruge systemet. To typer bruger har jeg bestemt at ville bruge systemet. En type som kun har rettigheder til at se status på anlægget(medarbejder) og en anden type som har lov til at skrive nogle beskeder i systemet(driftsleder).

Jeg har også prøvet at finde ud af hvordan de her bruger typer vil bruge systemet. Til det har jeg brugt use case til at analysere det. De use case som jeg har fundet, Log ind, Log ud, Se proces og indstillinger. Her under er de use case som jeg har defineret for systemet

Use case UC1: Log in

primary actor	Alle	
Goal in Context	At får adgang til renseanlægs hjemmeside	
Preconditions	Har adgang til systemet via mobiltelefon og browser	
main success scenario	Får adgang til alle tilgængelige informationer	
DESCRIPTION	**Step**	**Action**
	1	Indtast bruger navn
	2	Indtast bruger password
	3	Tryk ok/login
	4	**Kommer ind i system**
EXTENSIONS	**Step**	**Branching Action**
	1a	Forkert brugernavn System viser fejl og bruger får ikke adgang.
	2.a	**forkert password** 2.1 system signalere fejl og beder bruger at prøve igen.

Use case UC2: Log ud

primary actor	Alle	
Goal in Context	At logge ud fra systemet	
Preconditions	Er allerede inde i systemet via mobiltelefon og browser	
main success scenario	Logger ud fra systemet	
DESCRIPTION	**Step**	**Action**
	1	Tryk log ud
	2	Får bekræftelse at du ud af systemet.

Use case UC3: Se process

primary actor	Alle	
Goal in Context	Driftsfolk skal kunne se proces udvikling	
Preconditions	Skal være logget ind	
main success scenario	Driftsfolk skal kunne se om processen kører stabilt	
DESCRIPTION	**Step**	**Action**
	1	Vælg proces
	2	Vælge proces type
	3	Får vist processen
Extension	**Step**	**Branching Action**
		Log ud

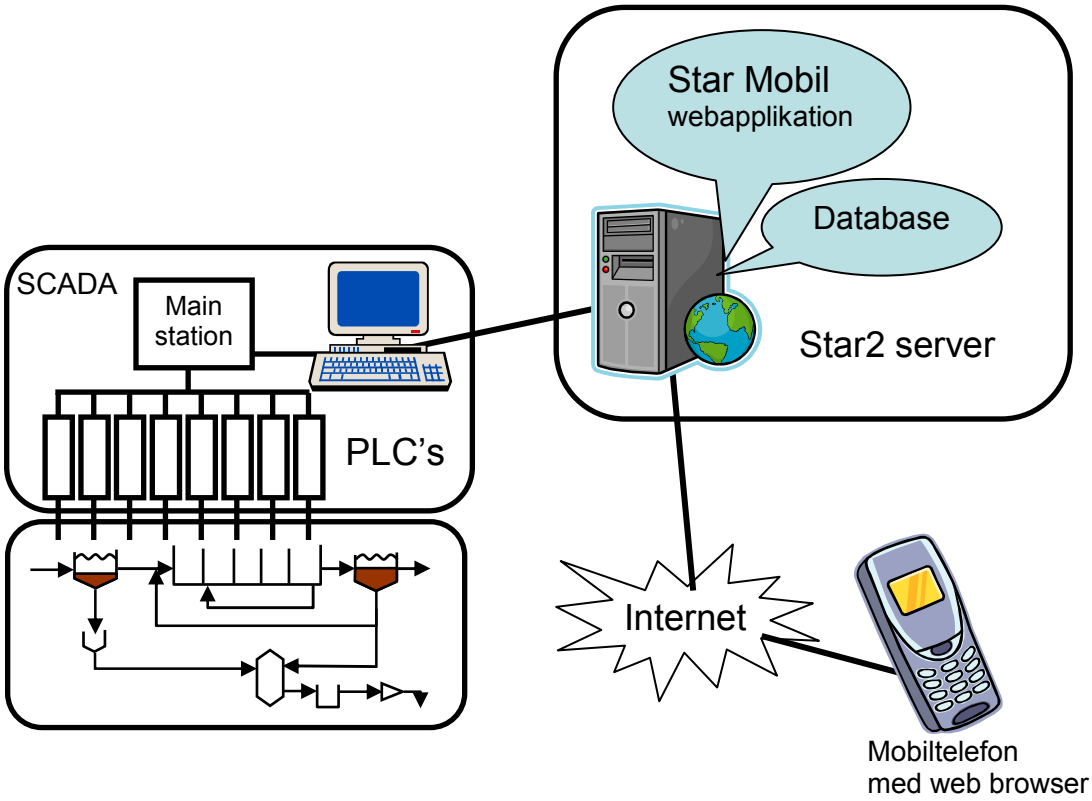
Use case UC4: Instillinger/Journal og Måler

primary actor	Driftsleder	
Goal in Context	En driftsleder skal kunne ændre nogle i processen	
Preconditions	Driftslederen skal være logget ind fra en mobil browser eller internet browser	
main success scenario	Ændre eller oprette noget i systemet	
DESCRIPTION	**Step**	**Action**
	1	Vælg "Journal" punktet i hoved menu
	2	Vælge journal som skal besvare
	3	Indtaste oplysninger som skal rettes
	4	Tryk ”ret”
	5	Tilbage til Journal
EXTENSIONS	**Step**	**Branching Action**
	4a	1.1 Annuller 1.2 Log ud

3 Design

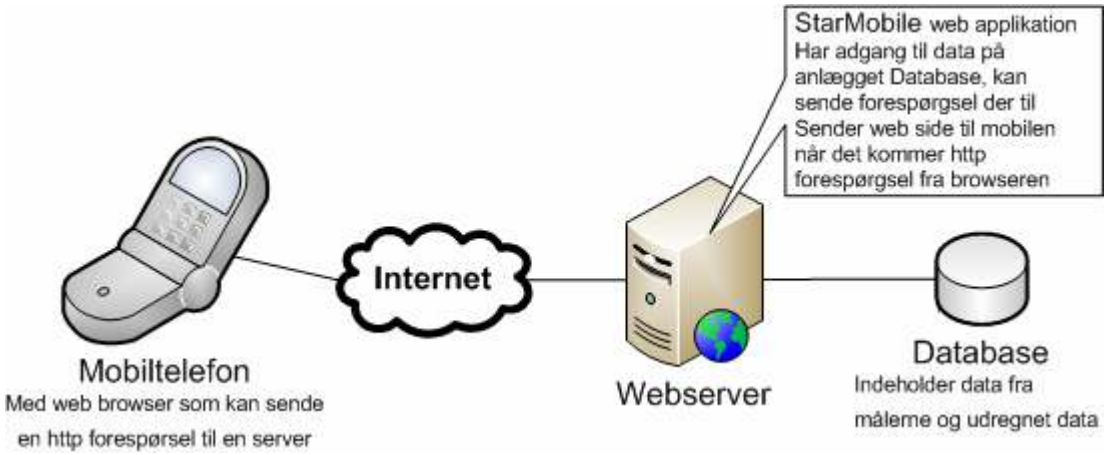
I denne kapitel vil systemet være designet til at opfylde kravene som var defineret i analyse fasen. Jeg vil starte at opsætte et overordnet billede af systemet og prøve så at design det i mindre bider ud fra den. Derefter vil jeg lave sekvens diagrammer ude fra use case. Til sidst er det beskrivelse af hvilken teknologi jeg har tænkt mig at bruge ved implementering af systemet.

3.1 Hoved struktur af systemet



Figur 5

Figur 5 danner et overordnet billede af systemet, den viser hvor min applikation vil ligge og hvordan brugeren kan få adgang til applikationen. Systemet vil være opbygget på den måde at det skal være et element(Mobil telefon, en PC) som har den mulighed at komme på Internet via en web browser til at få forbindelse til en web-server. Web browseren i telefonen kan sende en http forespørgsel til web server som har adgang til en database. Web serveren vil så kunne ud fra linket finde den rigtige web side og sende den tilbage til browseren.



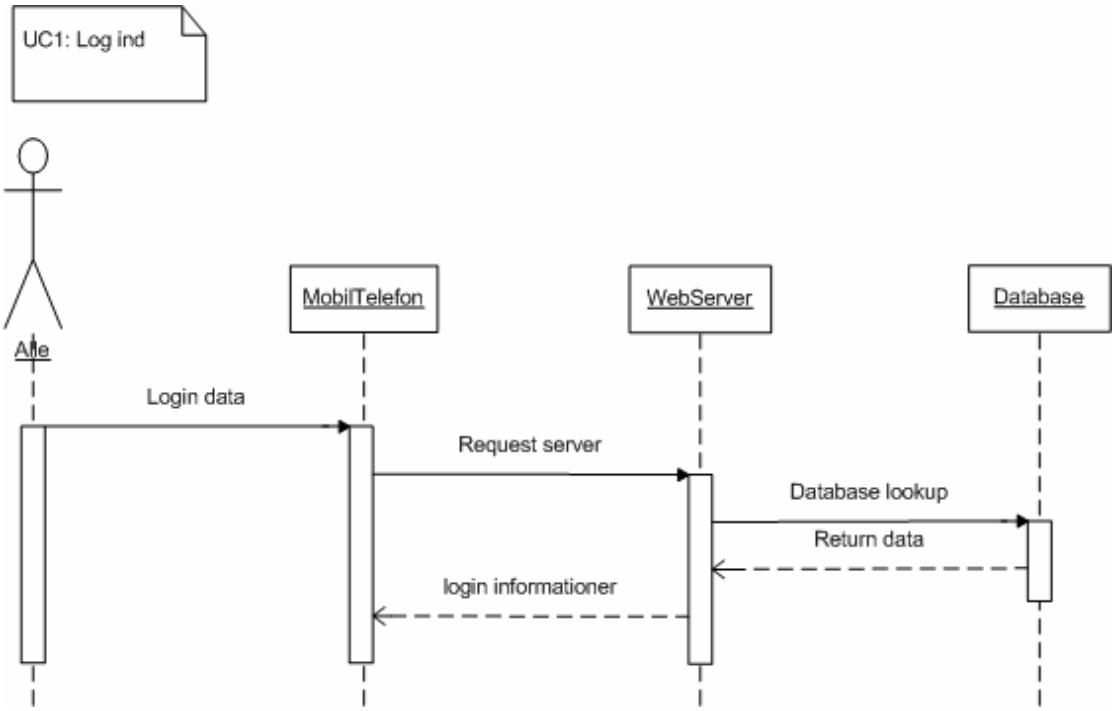
Figur 6

Applikation vil være opbygget på den måde at den vil ligge på en web-server som vil have adgang til data på anlægget database. Til at kunne få applikation vist, skal brugeren have adgang til den via en web browser. Web browseren vil sende en http forespørgsel til serveren om at vise StarMobile web applikation, f.eks. http://grock.star2.dk:28080/StarMobile/. Når serveren har modtaget forespørgselen vil den hente de data som er relevant for denne forespørgsel og sende svar tilbage med de informationer som brugeren sendt forespørgsel om.

3.2 Sekvens diagrammer

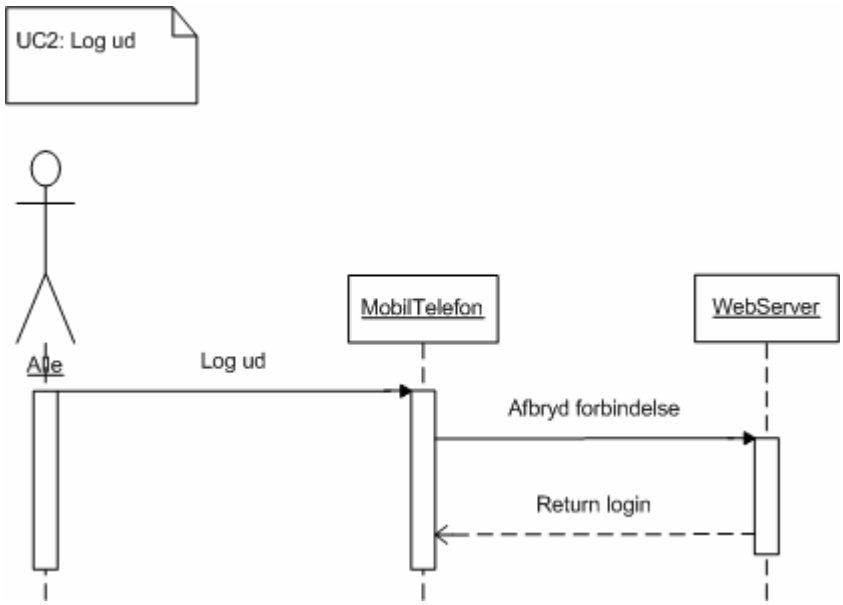
Ud fra use case har jeg lavet sekvens diagram for dem til at finde ud af hvordan strømmen på data vil være i systemet for hver enkelte use case. Med det fik jeg bedre forståelse på systemet og hvordan det skulle opbygges og implementeres.

Her er de sekvenser diagrammer for de use caser som jeg har lavet for systemet:



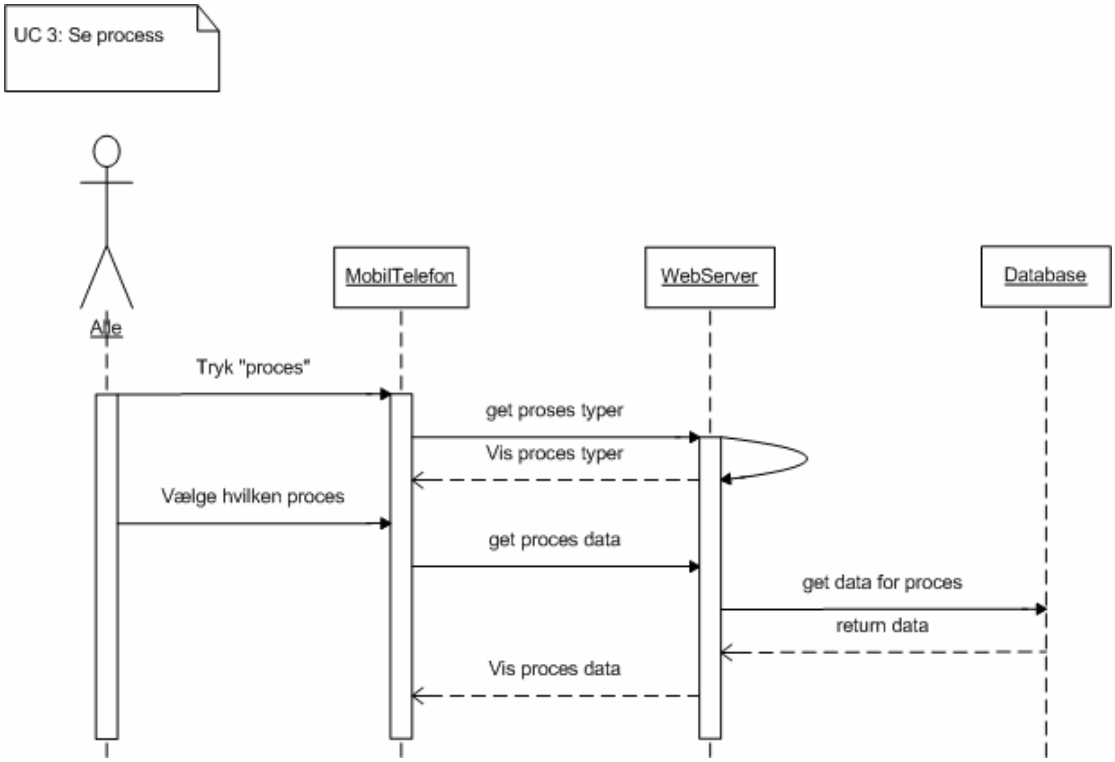
Figur 7

Når brugeren loger ind skal informationer sendes til serveren som tjekker om de er rigtige og serveren svarer tilbage om brugeren har adgang til systemet



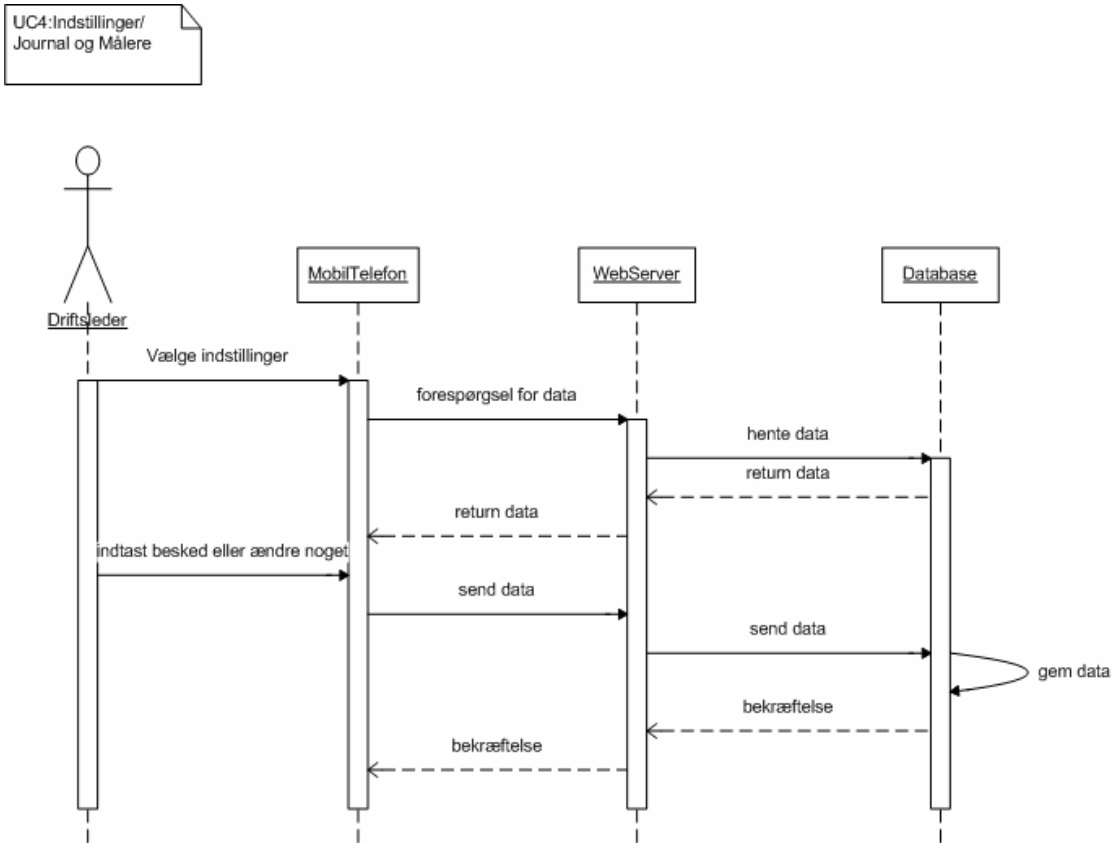
Figur 8

Når brugeren logger ud skal alle bruger oplysninger som serveren gemmer slettes.



Figur 9

Når brugeren trykker på proces/forside bliver sendt en forespørgsel til serveren om at hente processer i systemet. Serveren henter data fra en database og sender det til mobiltelefonen. Brugeren vælger hvilken proces han vil se og den proces bliver sendt til serveren som henter data fra databasen som så bliver sendt til mobiletelefonen

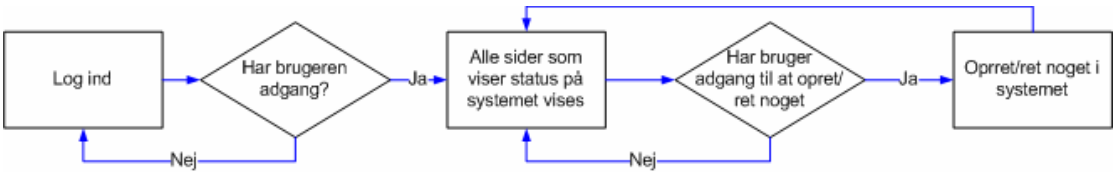


Figur 10

Når brugeren trykker på indstillinger/(Journal og Måler) sendes forespørgsel om data til serveren som henter data på databasen. Brugeren vælger hvilken data han vil ændre. Ændringen bliver sendt til serveren og den skriver den ned i en database.

3.3 Sikkerhed

Brugeren skal identificere sig selv med brugernavn og password til at log ind i hovedsystemet.



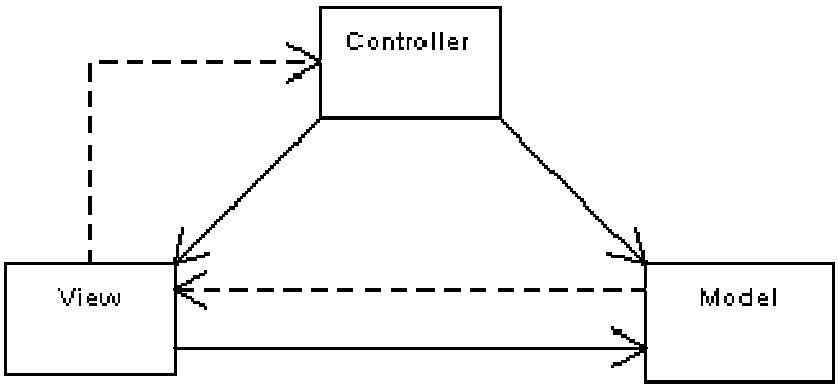
Figur 11

Når brugeren er logget ind vil han have adgange til de sider som han har rettighed til og nogle andre bruger vil have adgang til nogle ekstra sider som hører til deres bruger type. F.eks. en ”admin” bruger vil have mere rettigheder en almindelig

bruger, admin bruger vil have ret til at oprette noget hvor i mod en almindelig bruger ikke kan.

3.4 Arkitektur

Denne applikation har jeg valgt at designe efter Model-View-Controller Design(MVC) arkitekturen. MVC- arkitekturen bruges til at adskille applikation objekt(Model) fra hvordan det er præsenteret for brugeren(View) fra den mode brugeren kontroller det.(Controller)

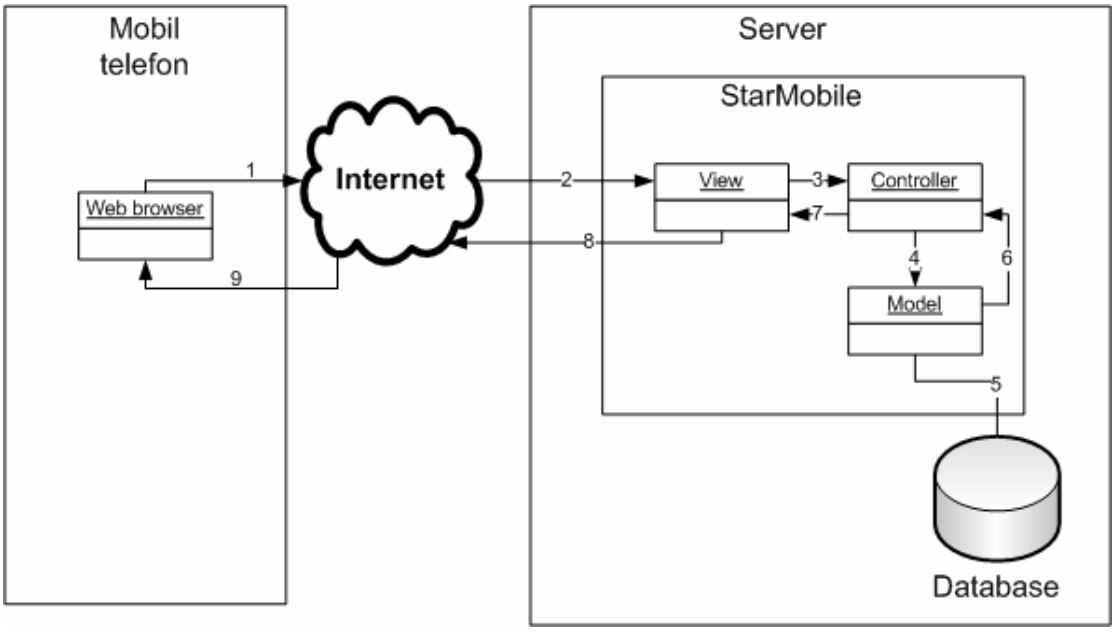


Figur 12

- **Model** kender alt om data som skal vises. Det kender også alt om alle operationer som kan bruges ved at flytte data. Men Model har ingen anelse om GUI, hvordan data bliver vist eller hvilken action GUI bruger til at manipulere data. Data er manipuleret og tilgængeligt i gennem metoder som er uafhængig af GUI. Model repræsenterer data og reglerne som gælder for adgang og opdatering af data.
- **View** holder styr på hvordan data bliver præsenteret for brugeren. View får status på Model fra kontrolleren. Det er View som har ansvar for at vedligeholde oplysningen når Model bliver opdateres.
- **Controller** objekt kender om den fysiske metode som manipuleret data det er sendt to model. Controller oversætter interaktion fra View ind i en aktion som skal udføres af model. I selvstændig GUI klient kan bruger interaktion være et klik på en knap eller noget valgt fra en menu, men ind i web applikation er en interaktion http anmodningerne GET og POST. Disse aktioner vil udføres af model inklusive aktivering af business proces eller ændring af model tilstands.

Baseret på brugeren interaktion og udkommen fra model aktion, vil Controller reagere med at vælge passende View. [litt. 15]

Jeg vil prøve så vidt som muligt at designe systemet efter MVC struktur og jeg vil opbygge systemet på følgende måde.

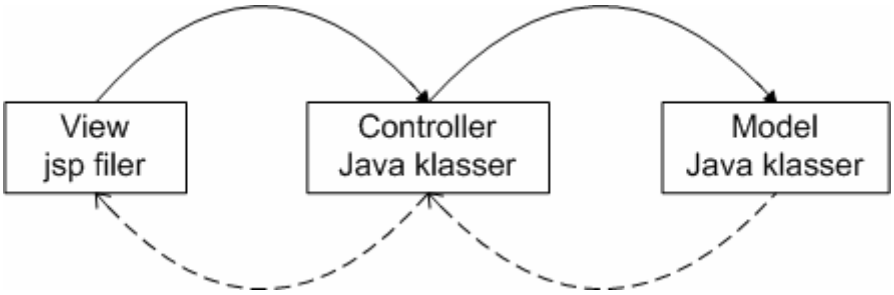


Figur 13

Her ses strømmen af data i applikationen.

1. Web browseren sender en forespørgsel til StarMobileud på Internet.
2. Server læser det og sender det til StarMobile applikationen. View modtager det.
3. View sender forespørgsel til controlleren om hvad skal vises på siden.
4. Controlleren sender forespørgsel til model om at hente data fra databasen.
5. Model henter data fra databasen.
6. Model sender data til controllen.
7. Controlleren sender til view hvad det skal vise på siden.
8. View sender siden til serveren, som siden sender det på nettet.
9. Web browseren viser siden.

Når systemet skal implementeres vil jeg implementere det på følgende måde:



Figur 14

- Alle view filer vil være af type jsp, og det eneste som de vil gøre er at hente data fra controlleren som skal vises på interfacet og sende en action til controlleren om det som brugeren gerne vil rette i systemet.
- Alle controller filerne vil være Java filer som kører på serveren det eneste som de gør, er at hente data fra model og sende de rigtige data til view.
- Alle model filerne vil også være Java filer som kører på serveren. Det eneste som de skal kunne, er at hente noget data fra database eller skrive noget data ned i databasen.

3.5 Design af brugergrænsefladen

Ved design af brugergrænsefladen vil jeg prøve så vidt som muligt at lave det sådan at brugeren godt kunne se hvorhen den står i systemet og gøre det nemt for brugeren at komme fra den en side til den anden. Alle siderne vil indeholde en bar i toppen, som skal være link til de forskellige områder, som er: **Forside**, **Proces**, **Målere**, **Journal**.

- **Forside** skal vise statussen på systemet som det er lige nu, det er data siden som skal vises. Der vil brugeren få hurtigt overblik om det er nogle alarm i systemet og hvilken fase det kører på anlægget. På siden vil brugeren have den mulighed at se de grafer som er mest interessant. Det vil være et link på siden som giver brugeren mulighed for at vise grafer og et andet link for brugeren at komme tilbage til data siden.
- **Proces** Her skal det være muligt for brugeren at kunne se de forskellige grafer og data for de **Proces** som bruges på renseanlægget. **Proces** siden vil indeholde en undermenu, som linker til de forskellige processer som kører på anlægget. På disse sider vil være mulighed for brugeren at vælge om han vil se data eller graf. Der vil være et link til data fra graf siden og omvendt.

- **Målere** Her vil det være muligt for brugeren at kunne se hvilke målere er aktive og hvem er ikke aktive, og hvornår de har sidst været ændret. Det vil også være muligt for brugeren at kunne slå en måler fra eller til.
- **Journal** Her vil det være muligt for brugeren at se beskeder fra andre og skrive nye beskeder ind eller svar beskeder fra andre brugere.

Ved opbygningen på brugergrænsefladen vil jeg prøve så vidt som muligt at opbygge det på samme måde som det er gjort i web udgaven af Star. Det vil jeg gøre så det vil være nemmer for brugeren at bruge mobil udgave af systemet, fordi han allerede kende alle komponenterne som bruges og hvilken betydning de har i systemet. Jeg vil prøve at have brugergrænseflade meget flydende og brugeren skal helst aldrig være i tvivl hvor han står i systemet. Brugeren skal helst kunne kigge på skærmen og se med det samme hvad han kigger på. Det skal også være meget nemt for brugeren at komme fra den ene side til en anden uden at det kræver at brugeren husker vejen der til.

Alle siderne vil være opbygget på samme måde og funktionalitet skal være ens i gennem hele applikationen, dvs. hvis der skal aktivere noget skal det helst ske på samme måde alle steder.

Mest af teksten som står på siderne skal læse fra XML filer. I dem kan man så definere hvor mange komponenter det vil være interessant at se på mobilen, hvilken label de skal have og hvorhen de data ligger som skal bruges for den komponent. Det vil gøre det nemmere at bruge applikationen på flere en et anlæg. Fordi det eneste som skal ændres er XML filerne, med at skrive de informationer som passer for anlægget.

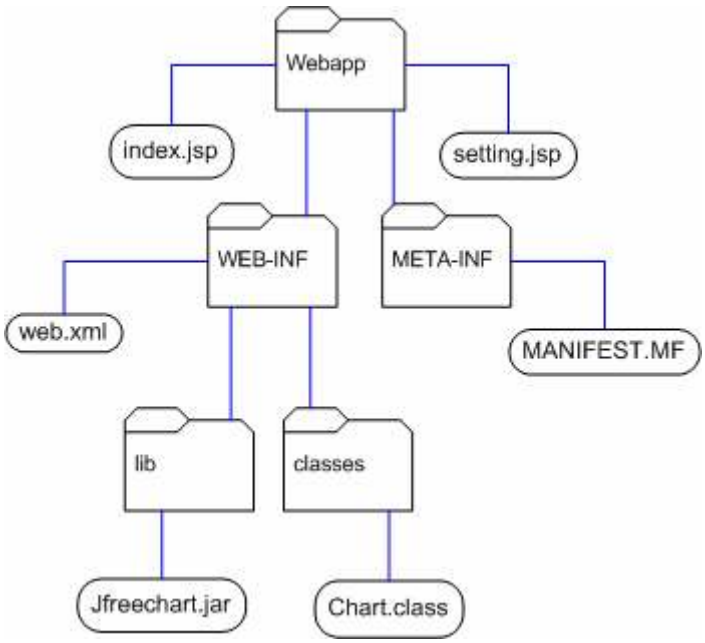
3.6 Importere applikationen

Til at importere applikationen på serveren vil jeg lave en war fil som jeg flytter over på web serveren. War filen vil indeholde alt om applikationen og hvordan den skal fungere. Til at lave en war fil vil jeg lave en ant script til at opbygge en war fil. Bagefter vil jeg skrive om teknologien som jeg vil bruge ved at importere applikationen. Her er beskrivelse af en war fil og hvordan den skal være opbygget.

3.6.1 War filer (Web Archive)

Web applikation er en gruppe af HTML sider, JSP sider, ”servlets”, ”resources” og ”source” filer, som kan være styret som en enkelt enhed. En ”Web Archive” fil er et pakket web applikation. WAR fil kan bruges til at importere en web applikation til en web server. [litt. 21]

3.6.1.1 Opbygning på en war fil



Figur 15

Web applikation (war) har prædefineret adressekatalog struktur som fastslår hvor klasses, bibliotek og generel ressources skal ligge. Med denne struktur er det nemmer for serveren at behandle de forskellige web applikationer.
Figur 15 viser adressekatalog strukturen på en web applikation.

- **webapp:** er rod mappen i strukturen. Den indeholder generel ressources som html sider, jsp sider osv. Det er anbefalet at lave undermapper under roden til at have bedre organisation.
- **webapp/WEB-INF:** indeholder web applikation deskriptor filen, web.xml
- **webapp/WEB-INF/classes:** her er alle klasse filer placeret som bruges i web applikationen. Det er rod katalog for alle klasse ”package”.

- **webapp/WEB-INF/lib:** her er alle hjælp biblioteker(JAR filer) placeret som bruges. Alle JAR filer som findes i den katalog er automatisk sæt ind i classpath. Servlet kan pakkes i en JAR fil og blive lagt i den katalog.
- **webapp/META-INF:** indeholder MANIFEST.MF

Man kan selv vælge om filen i war filen er direkte tilgængelig fra ydersiden af webapp. Hvis de skal være tilgængelig fra yderside så skal de ligge direkte under webapp mappen, eller i nogle undermappe som udvikleren selv har oprettet. Men hvis der er noget som ikke skal være direkte tilgængelig fra ydersiden så skal de filer ligge i WEB-INF eller i META-INF, fordi alt under WEB-INF og META-INF er ikke direkte tilgængeligt. Hvis serveren bliver spurgt om noget som ligger under WEB-INF elle META-INF, f.eks. *http://grock.star2.dk:28080/StarMobile/WEB-INF/web.xml*, skal den svare med ”404 NOT FOUND”. Alle filer som ligger under webapp har adgang til filerne som ligger i WEB-INF og META-INF, derfor ligger alle hjælpefiler som bruges i web applikationen derunder. [litt. 1]

3.6.1.2 Hvilken betydning har web.xml for en war fil

Når man indsætter et web applikation i container på serveren, skal der laves et meget simple XML dokument(web.xml) som hedder ”Deployment Descriptor-(DD)”. DD fortæller din container hvordan det skal køre dine servlets og JSP. Hoved opgaven for web.xml er at definere ”mapping” mellem den virkelig web adresse som brugeren bruger og den aktuel adresse på ”servlet” klassen

I web.xml filen kan man definere mange forskellige konfigurationer som web applikation containeren vil bruge. De kan f.eks. være konfiguration for *authentication*, fejl sider, filter, sikkerhed, session tidsudløb (timeout) osv.[litt. 1] Jeg skriver mere om hvordan jeg bruger det i min applikation i implementering delen af rapporten.

3.6.2 Hvad er en Container?

Container er et Java applikation. Når en web server applikation får en forespørgsel om et servlet, sender den forespørgselen videre til containeren, men ikke til servlets selv hvor servletet er anvendt(deployed). Det er så containeren som sender http forespørgsel(request) til og svar(response) fra servletes selv og kalder på servletes

metode(doPost og doGet()). Containeren kan selv finde ude af hvordan den skal kommuniere med web serveren via en protokol, så det eneste som en udvikler skal tænke om er udviklingen på en applikation.

Containeren holder styre på hvor længe hvert servlet skal holdes i live. Den tager sig af at loade klasserne, instantiering og initialisering af servlet, kalder på servlets metode og laver servlets ”instance” valgbar for ”garbage collection”.

Containeren laver automatisk nye Java tråde for hvert servlet forespørgsel (request) den modtager. Når servlet er færdig med at køre HTTP service metoden for klienten forespørgsel, dør tråden. [litt. 1]

3.6.3 Apache Ant

Apache Ant er et software værktøj til automatisk styring af compilering og pakning. Ant er skrevet i Java sproget, kræver Java platform og er bedst velegnet til at bygge Java projekt. Ant bruger XML til at beskrive bygge processen og hvilken target er afhængig af hinanden. Ant XML filen har default navnet build.xml. (se Bilag G.3) [litt. 19]

3.6.4 JavaServer Pages/Servlet

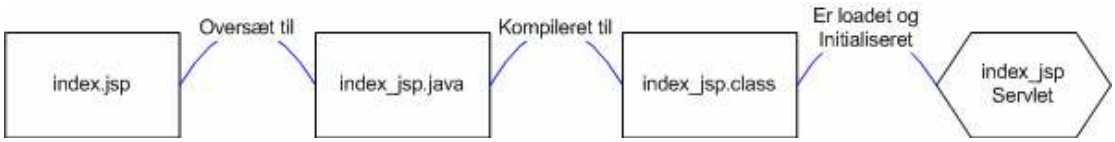
Servlet er et Java program som skal installeres på web serveren, efter at det er gjort kan det modtage forespørgsel over Internettet. De vil typisk være fra http protokollen.

Der er tre metoder som kan overskrives når der skal programmeres et servlet; de er init, destroy og service.

- *Init* metoden bliver kaldt når servlet er oprettet.
- *Destroy* metoden bliver kaldt ligeför servlet bliver deaktiveret og fjernes.
- *Service* metoden bliver kaldt hver gang servletet får en forespørgsel.

Hvis der bruges http protokol er det en fordel at bruge en af HTTPServlet service metoder, som svarer til de forskellige forespørgsler typer i HTTP protokollen. [litt. 6]

En ”Java Server pages”(jsp) er et html side med en Java kode som kører på siden. Det giver den fordel for udvikleren at den kan skabe dynamiske hjemmesider.



Figur 16

En jsp fil i en web applikation bliver til et servlet når den bliver lagt ind på containeren(serveren). Containeren tager jsp filen og oversætter den til en servlet klasse kilde(source)(.java) fil, som til sidst bliver kompileret til en servlet klasse. Bagefter virker det som et almindelig servlet, som en udvikler selv har skrevet. [litt. 1]

3.6.5 Indlæsning på XML dokumenter

DOM(Document object model) er API(application programming interface) for velformet XML dokument og gyldige HTML fil. DOM definere den logiske struktur af dokumentet, hvordan det kan blive manipuleret og hvordan der gives adgang til det. XML bruges som repræsenterer af forskellige informationer som vil så blive gemt i mange forskellige systemer, og mange af dem vil blive set som data men ikke som dokument. Men alligevel vil XML præsentere data som dokument og DOM bruges til at håndtere de her data.

Med DOM kan programmet lave et dokument, læst dokument struktur, og tilføje, ændre eller slet elementer og indhold i dokumentet. Til at få adgang til XML og HTML kan man brug DOM, men med få undtagelse. [litt. 7]

DOM interfacet er meget let at forstå. DOM analysere hele XML dokumentet og opfører repræsentation af det i hukommelsen med hjælp af klasse model konceptet som findes i DOM.

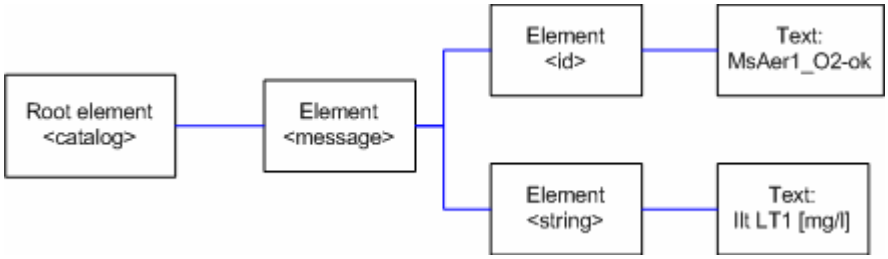
DOM analyser er kaldt ”DocumentBuilder” i Java, fordi den bygger en repræsentation af dokumentet i hukommelsen. DocumentBuilder skaber en instans af org.w3c.dom.Document som er en træ struktur da indeholder knuder fra XML dokument. Hver knude i strukturen implementerer så en org.w3c.dom.Node interface. Der findes mange forskellige knuder der repræsenter data fra en XML dokument. De vigtigste knuder type er:

- Element knuder, kan indeholde attributter
- Tekst knuder, repræsenterer teksten som står i mellem start og end tag af et element. [litt. 8]

Her kan man se eksampel af et XML dokument:

```
<catalog>
      <message>
            <id>MsAer1_O2-ok</id>
            <string>Ilt LT1 [mg/l]</string>
      </message>
</catalog>
```

XML dokumentet oven på vil blive opsat på følgende måde af DOM objektet:



Figur 17

3.7 Værktøjer som skal bruges ved implementering

De værktøjer som jeg vil bruge ved udviklingen på applikationen er

- *Eclipse*
- *Java*(jdk1_5_09)
- *ant*(apache-ant-1.7.0)
- *Evrsoft First Page 2006*
- *CSS Tab Designer*
- *Internet Explorer*
- Web browser i en mobile telefon, *Nokia E60*. Browseren skal kunne understøtte JavaScript og stylesheet.
- Jeg har haft adgang til en *MySql* database og en *JBoss* server hvor jeg har lagt min applikation ind. Web-serveren skal kunne aktivere war fil og have adgang til J2SE 5 eller højere.

4 Brugergrænsefladen

I denne kapitel vil jeg forklare de forskellige brugergrænseflader som jeg har implementeret i min applikation og vise nogle skærmbilleder af dem.

Det første skærmbillede som vises for en bruger når han har fået adgang til systemet kan ses på figur 18. Øverst er hovedmenuen med de fire faneblader.

Figur 18

Menuen er på alle sider. Fanebladet som er aktivt har hvid farve på teksten og hvide linier kommer ud fra fanen. Hvis brugeren kører muse markøren over de andre faneblader, bliver teksten hvid og fanen lidt større. Hvert faneblad har link til de sider som de står for.

Figur 19

Figur 19 viser når muse markøren er placeret på **Målere** fanebladet og hvis brugeren klikker på den bliver **Målere** fanebladet vist.

Figur 20

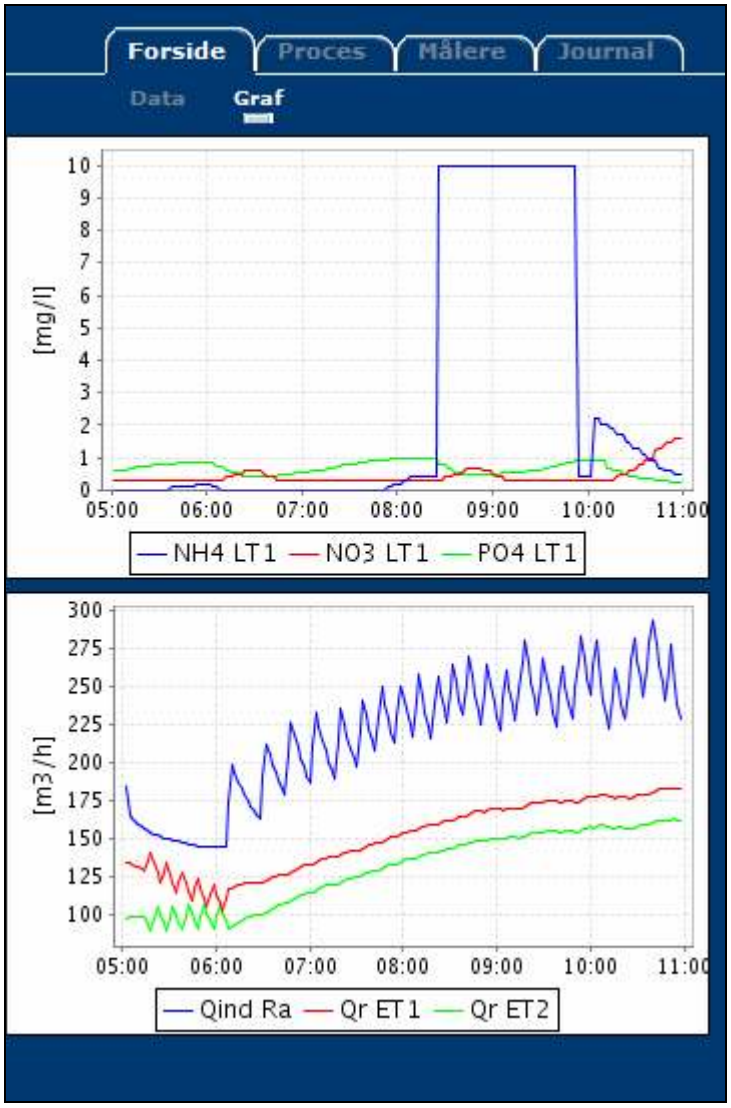
For under menuerne er det samme princip som for hovedmenuen. På figur 20 er Fase proces aktiv og muse markøren er over Returslam linket og hvis brugeren klikker på det bliver den proces vist. Det samme gælder for Data-Graf menuen.

Figur 21

Her er muse markøren over Data linket med Graf linket aktivt.

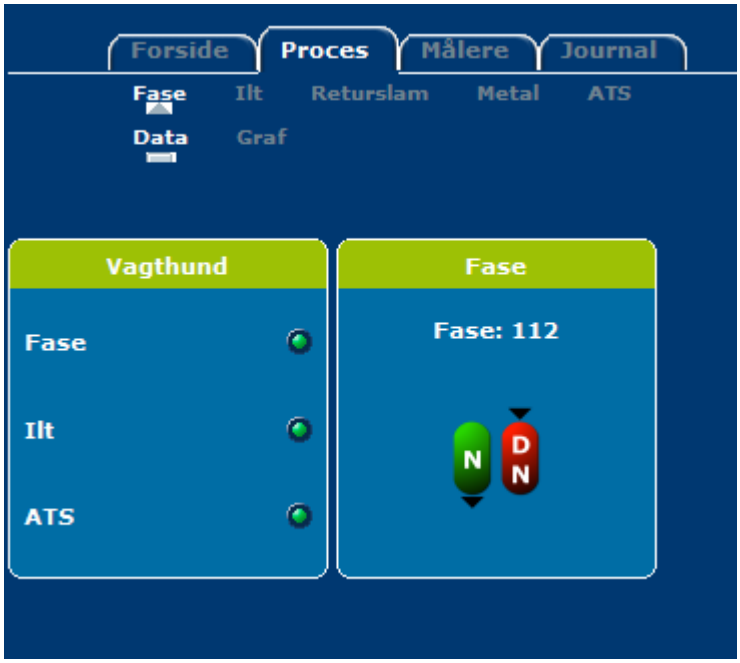
Informationerne som vises på start siden er statusen på anlægget lige nu, se figur 18. Data som ses præsenterer:

- **Vagthund**, kan vise rød, gul eller grønne lamper, som siger til om tilstandet på STAR og SRO
    a) **Rød** lampe, så fejler begge Star og SRO
    b) **Gul** lampe, så er det enten Star eller SRO som fejler.
    c) **Grøn** lampe, så er begge to er OK.
- **Fase**, viser hvilken fase kører i luftningstanken
- **Data kvalitet**, om opsamlede data er ok eller ej
- **ATS**, nedbør udsigt, hvor meget regn kan forventes til at uden gå at slam ikke skylles ud i naturen.



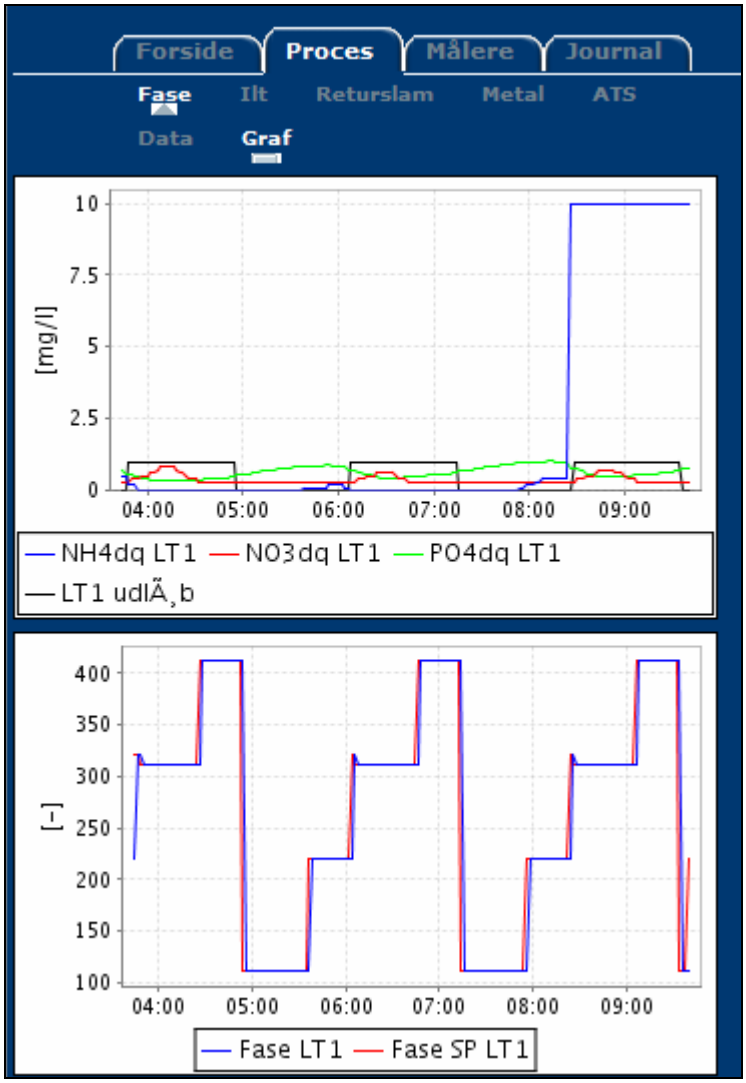
Figur 22

På figur 22 kan brugeren se de grafer som er defineret at skulle vises på **Forside**. Det er vigtigste proces grafer i systemet.



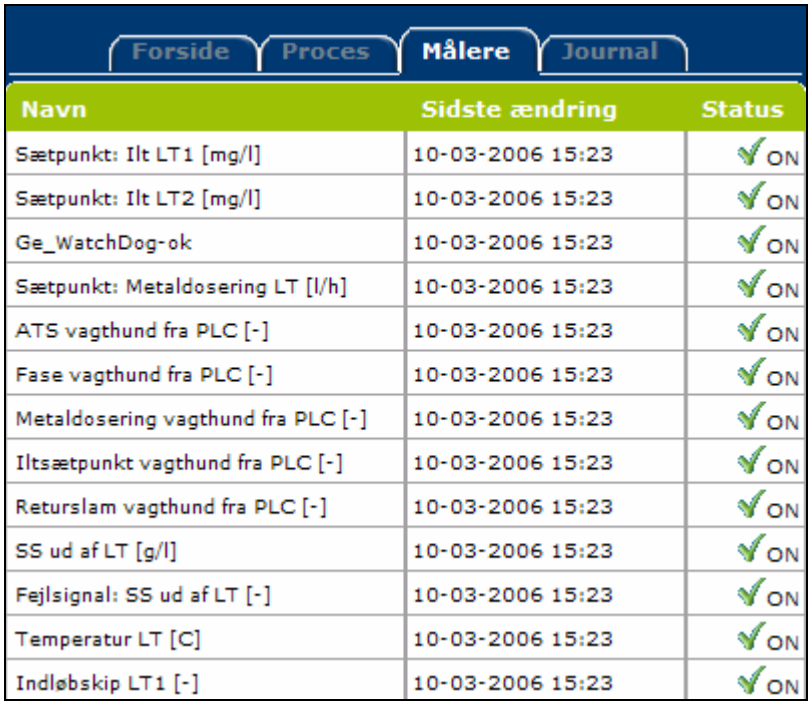
Figur 23

På figur 23 ser brugeren data for Fase proces



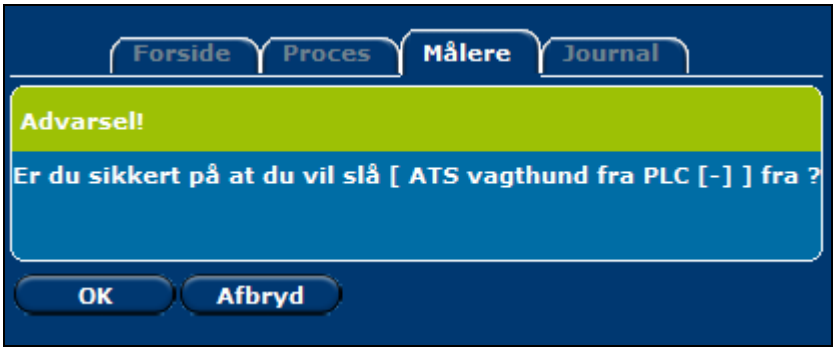
Figur 24

På figur 24 ses graferne for Fase Proces



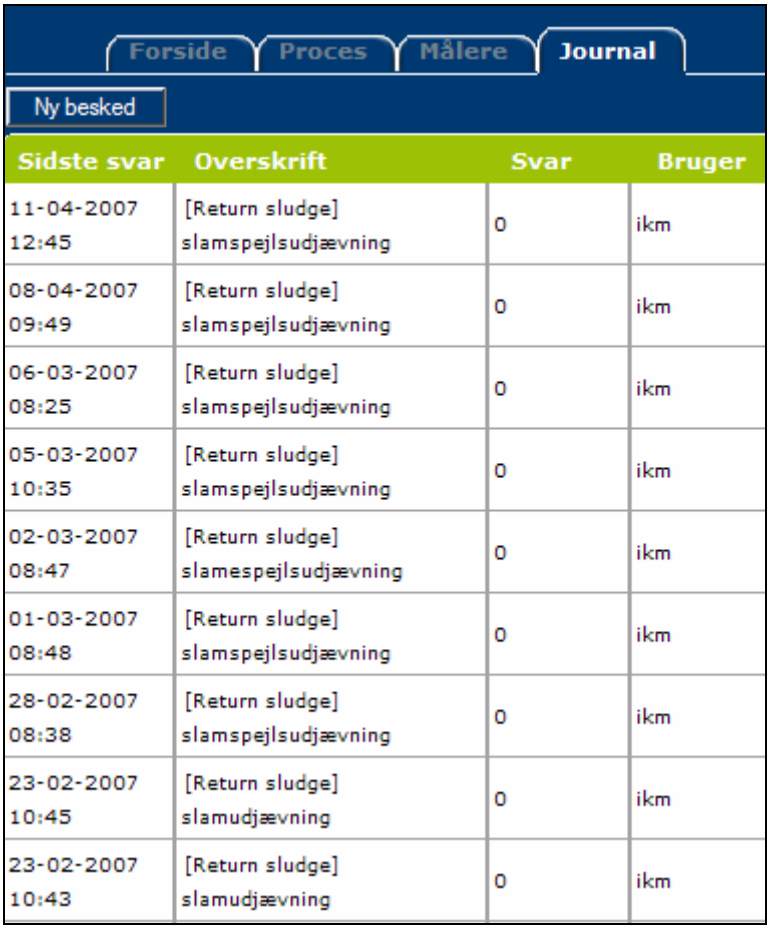
Figur 25

På figur 25 vises alle målerne som bruges i systemet. Brugeren kan vælge at slå en måler fra med at vælge målerens navn. Navnene på målerne har et link til side som ses på figur 26



Figur 26

På figur 26 kan brugeren vælge om at slå den valgte måler fra. Til at slå en måler fra skal brugeren have rettigheder til det.



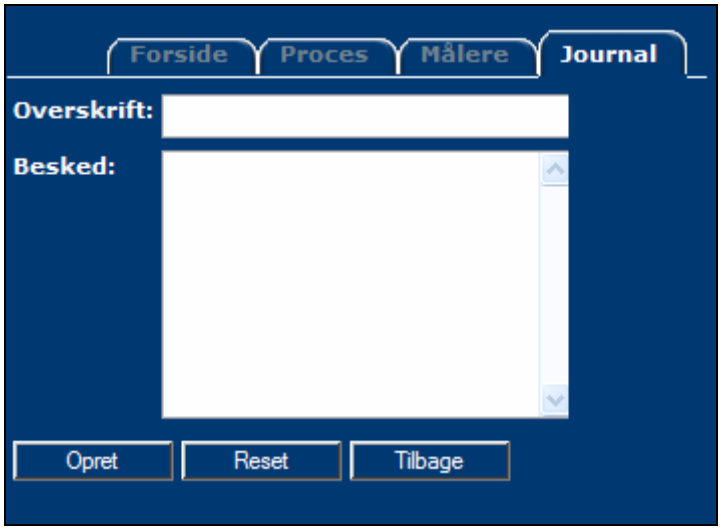
Sidste svar	Overskrift	Svar	Bruger
11-04-2007 12:45	[Return sludge] slamspejlsudjævning	0	ikm
08-04-2007 09:49	[Return sludge] slamspejlsudjævning	0	ikm
06-03-2007 08:25	[Return sludge] slamspejlsudjævning	0	ikm
05-03-2007 10:35	[Return sludge] slamspejlsudjævning	0	ikm
02-03-2007 08:47	[Return sludge] slamespejlsudjævning	0	ikm
01-03-2007 08:48	[Return sludge] slamspejlsudjævning	0	ikm
28-02-2007 08:38	[Return sludge] slamspejlsudjævning	0	ikm
23-02-2007 10:45	[Return sludge] slamudjævning	0	ikm
23-02-2007 10:43	[Return sludge] slamudjævning	0	ikm

Figur 27

På figur 27 vises alle journaler i systemet. Brugeren kan vælge at læse den ved at klikke på Overskriften af journalen og vises så skærmbilledet ligesom på figur 28. Brugeren kan også vælge at lave ny journal en med at klikke på ”Ny besked” og vises så skærmbilledet lige som figur 29.

Figur 28

På figur 28 ser brugeren beskeden som han har valgt at læse. Brugeren kan vælge at svare dem eller skrive ny besked.

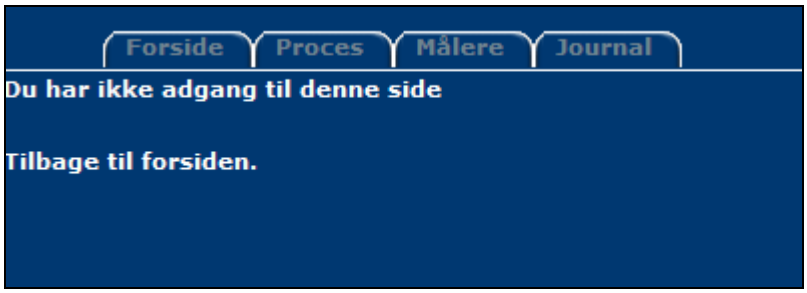


Figur 29

På figur 29 kan brugeren skrive ny besked. Med at klikke på Opret bliver beskeden gemt. Til at skrive ny besked skal brugeren have rettigheder til det.

Figur 30

På figur 30 kan brugeren vælge at svare gamle beskeder. Med at klikke på Opret bliver beskeden gemt. Til at svar besked skal brugeren have rettigheder til det.

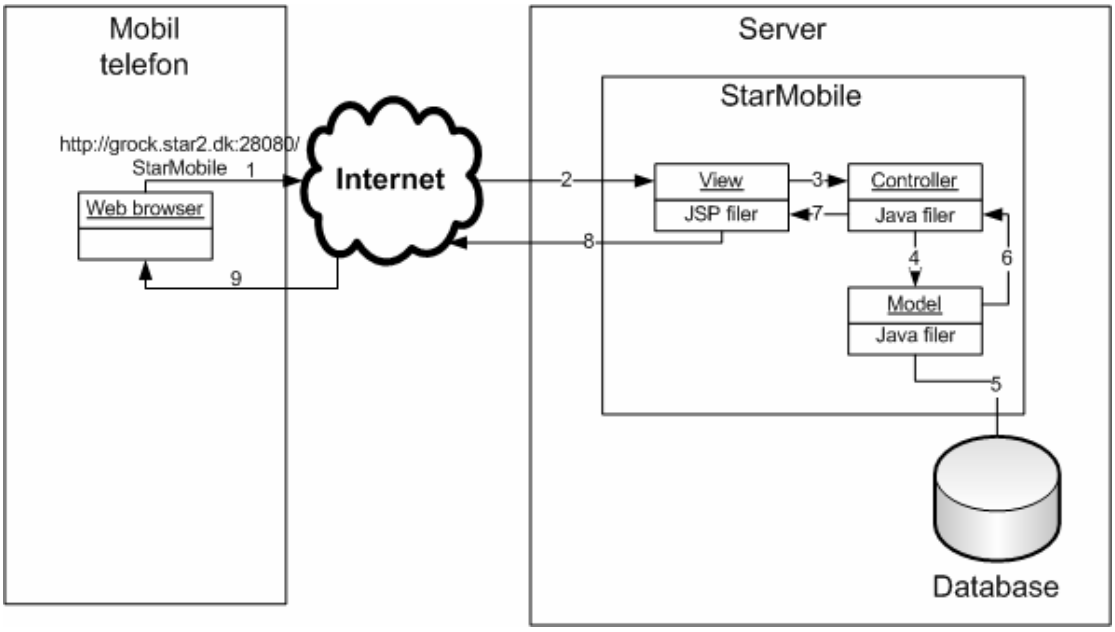


Figur 31

Figur 31 vises når en bruger som ikke har rettigheder til at oprette eller ændre på noget i systemet

5 Implementering

I denne kapitel vil jeg forklare hvordan jeg har implementeret applikationen, hvilken klasser jeg har implementeret og hvordan de kommunikere sammen. På figur 32 vises struktur billede af systemet som giver godt overblik over virkningen i systemet og strømmning af data fra en komponent til en anden. Billedet viser mere informationer en Figur 13 i Design fasen, fordi den viser mere hvordan jeg har implementeret mit system.



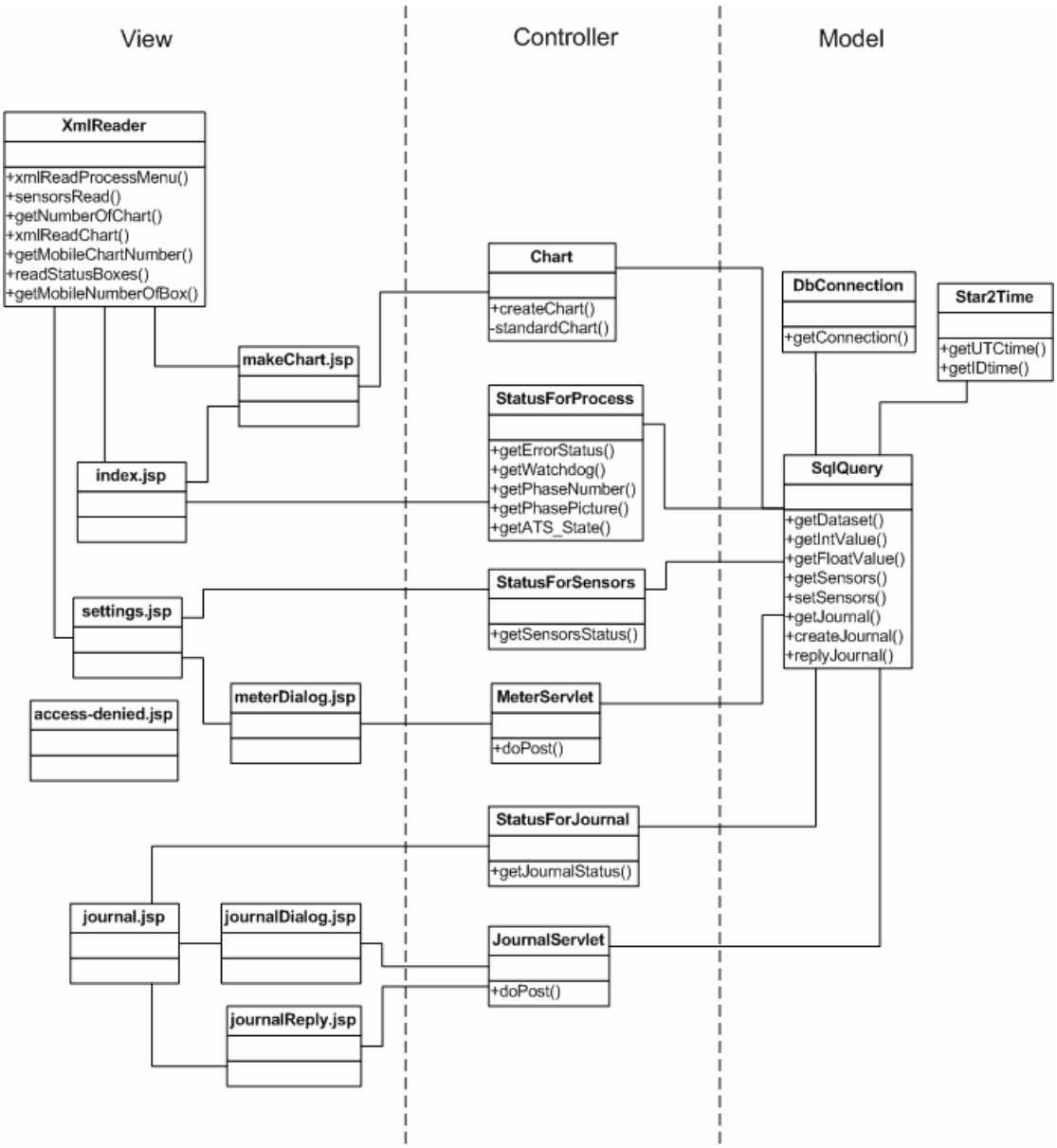
Figur 32

På figur 32 kan det ses hvordan en forespørgsel fra en hjemmeside kommer ind på serveren og ind i applikationen.

1. Web browseren sender en forespørgsel til StarMobile ud på Internet
2. Server læser forespørgselen og sender den til StarMobile applikationen. View modtager den med en jsp fil
3. View sender forespørgsel til Java klasse som er en controller, om hvad skal vises på siden
4. Controlleren sender forespørgsel til Java klasse i modelen om at hente data fra databasen
5. Modellen henter data fra databasen
6. Modellen returnerer data til controllen
7. Controlleren returnerer til view hvad det skal vise på siden
8. View sender html kode til serveren som sender det på nettet

9. Web browseren læser koden og viser resultatet

Ved implementering af applikationen har jeg prøvet at holde fast i den MVC designe struktur som jeg har skrevet om i design kapitel(3.4 Arkitektur), dvs. MVC. Alle jsp filerne styre kun brugergrænsefladen og henter alle sine data oplysninger til controlleren, som er Java filer som igen henter alle data fra en model som har adgang til en database. På figur 33 ses hvordan jeg har opbygget min applikation og hvilke klasser der er implementeret. Tavle over filer navnerne kan ses i Bilag B og liste over metoderne som er blevet implementeret i Bilag C.



Figur 33

Da jeg startede at implementere applikationen begyndte jeg på at få et system til at køre og virke som jeg har skrevet i Design. Da fik jeg mange filer som lignede

hinanden, lavede det same men med forskellige parameter og skulle hente data fra forskellige steder. Efter at jeg havde fået strømningen i applikationen, fra web browser til web server og omvendt, prøvede jeg at lave min kode således at den kunne genbruges så meget som muligt, og at det ikke vil være dublering af koden som vil gøre det sværere at vedligeholde i drift. Til at kunne genbruge koden har jeg forsøgt at gør den dynamisk, dvs. startede at sætte alle filerne som lignede hinanden i en fil(index.jsp) og denne fil skal bare få de parametre med i forespørgsel som vil fortælle hvilken interface det skal vises og hvilken oplysninger/komponenter det skal vises på interfacet. Koden for applikationen er i bilag, den er delt ned i view kode (Bilag D), controller kode (Bilag F)og model kode (Bilag E).

Ved implementeringen af applikationen brugte jeg både en almindelig browser og en browser i en mobiltelefon. Det er vigtigt at bruge browseren i en mobiltelefon til at få den rigtige størrelse af siderne og til at se om den kunne klare den teknologi som jeg vil prøve at bruge. Mobilbrowseren var lidt langsom i brug, derfor har jeg også brugt en almindelig browser. Den brugte jeg mest når jeg skulle tjekke virkningen i applikationen. Der var nogle gang hvor mobilebrowseren ikke kunne lave det som en almindelig browser kunne, f.eks. at linke et billede på en knap.

5.1 Implementering af brugergrænsefladen

Brugergrænsefladen er implementeret med JavaServer pages, dvs.i jsp filer. I den er det muligt at skrive html kode, bruge Java kode og klasser som har været implementeret i systemet, og det har jeg gjort i min implementering af brugergrænsefladen.

Jeg har brugt stylesheet til at beskrive hvordan jeg vil at min html kode bliver repræsenter i web browseren. Med stylesheet kan jeg kontrollere hvordan siderne ser ud og opbygningen på dem.

Når jeg var i gang med at implementer interfacet brugte jeg to stylesheet fra Star control koden(*box.css og list.css*) og en af dem har jeg selv lavet(*tabStyle.css*) (bilag H.1) med hjælp af en stylesheet designer[litt.17]. De to stylesheet fra Star control har jeg valgt at bruge til at få de samme komponenter (boks og tavle) som er i Star control. De stylesheet laver boxerne(på **Forside** og **Proces**) og tavle listen som kan ses under **Måler** og **Journal** faneblade. Jeg har selv måtte finde ud af hvordan det bruges i Star control fordi at de som arbejder i Krüger har ikke selv lavet

brugerinterfacet men de har fået et firma til at lave det for dem og det er mest skrevet i JavaScript. Det stylesheet som jeg har lavet, er det som styrer menuerne og undermenuerne. Det fortæller hvordan et link skal se ud(hvilket billede skal med), hvad den skal gør mens den er aktiv og hvordan den skal regere når musen markøren kører over linket.

Stylesheet kan implementeres i head taget i html koden. I stylesheet kan man definere hvordan en html tag skal opføre sig når det bliver vist i browseren. Til at definere hvordan det enkelte tag skal være, kan attributten id blive sætte til navnet på scripten i stylesheet. Til at tilføje id bruges ofte tag * *, her har jeg skrevet at det skal bruges scriptet fra *tabJ* for alle tag som er blevet defineret i stylesheet og er defineret i mellem de to tage.

Alle statiske billeder som vises på brugergrænsefladen er de samme som bruges i Star control, jeg har fået lov at kopiere dem ind i mit projekt så jeg kunne vise dem på brugergrænsefladen.

Jeg har kun lavet en fil *index.jsp* som viser **Forside** og **Proces** fanebladene. Fordi at jeg bruger jsp ved implementering af brugergrænsefladen kan jeg hente nogle parameter i linket og tjekket dem i en meget simpel if else statement, der med fundet ud af hvad skal vises. Parameterne som jeg bruger fra linket hedder *pageName*, p*rocessName* og *processType*.

- Parameteren *pageName* indeholder oplysninger om det skal vise **Forside** eller **Proces** fanebladene, hvis den er tom så bliver **Forside faneblad** vist.
- Parameteren *processName* bruges kun for **Proces** faneblad. Den finder ud hvilken proces brugeren nu vil kigge på, hvis *processName* er tom, og *pageName* er process så bliver **Fase** vist under **Proces**.
- Parameteren *processType* siger til om det skal vise data eller graf på siderne, både **Forside** og **Proces** bruger denne parameter. Hvis den er tom så bliver data vist.
    - Hvis proces type er data bliver nogle Java metoder fra controleren kaldt til at finde ud af hvad der skal stå i bokserne og hvilke billeder der skal vises.
    - Hvis proces type er graf bliver størrelsen på skærmen hentet med JavaScript, til at definere bredden på graferne. Bagefter bliver så makeChart.jsp kaldt med nogle parametre, se forklaring i kaptiel 5.2.

Nu skal der undersøges hvordan *index.jsp* arbejder når den får en forespørgsel som *http://grock.star2.dk:28080/StarMobile/index.jsp?pageName=process&;*

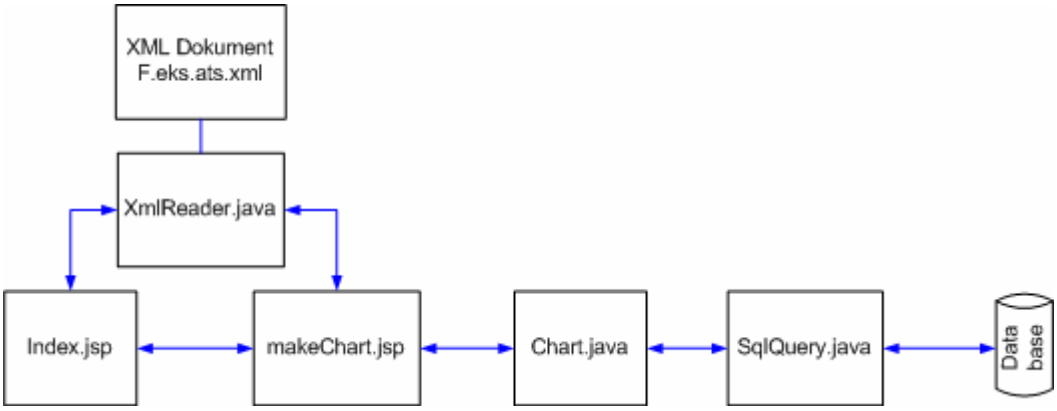
processName=process/phase&;processType=graph&;

1. Det første som sker i *index.jsp* er at alle parametrene er hentet og bliver gemt i variables. Hvis nogen af parametrene var tom bliver de sat til default værdier.
2. Nu bliver tjekket om det er **Forside** eller **Proces** faneblad som skal være aktivt. Det er gjort med at tjekke pageName og hvis den er sat til proces så skal under menu på **Proces** fanebladet hentes. I linket står det at det skal være **Proces** faneblad som skal være aktivt
3. Her bliver så *processName* læst til at se hvilken proces skal være aktiv i menuen. I linket står at det skal være processen **Fase**.
4. Her bliver det afgjort om det skal vise data eller grafer på siden. Dette er tjekkes med at læse *processType*, hvis den er data bliver **Data** vist på side og data linket bliver aktiv, men hvis den er graph bliver **Graf** vist på siden med graf linket aktiv. I linket ovenpå står at **Graf** linket skal være aktivt.

For **Journal** fanebladet har jeg også lavet en jsp fil(*journal.jsp*) ligesom for hovedsiden og den side som viser hver enkelte journal. Der er kun en parameter sendt med til at finde ud af hvad det er brugeren vil se. Parameteren hedder *journalID* og hvis den er sat til et værdi så vil der være vist et enkelte besked, men ellers vil der være vist hovedside for journal med tavle for alle journal. Jeg har lavet to andre jsp filer, en for at lave ny journal(*journalDialog.jsp*) og en for at svare besked(*journalReply.jsp*).

Målere fanebladet har tavle med alle målerne i systemet hvor der ses om den er aktiv eller ej. Filen som det er implementeret i er *settings.jsp*. Når der skal ændre status på en måler, skal brugeren vælge måleren derefter bliver brugeren sendt til en anden side(*meterDialog.jsp*) med parameterne *sensorName*, *sensorID* og *sensorStatus*. Parametrene hjælper applikationen at finde hvilken sensor det skal arbejde med.

5.2 Hvordan bliver graferne lavet



Figur 34

- *index.jsp* skal vise et graf på siden, bliver sendt en forespørgsel til *makeChart.jsp*. Forespørgselen indeholder bl.a. parameter for bredde og højde på siden, *screenWidth* som har samme værdi og skærmstørrelsen på mobiletelefonen. Parameteren *screenWidth* definerer bredden på grafen og *screenHeight* definerer højden på grafen. De andre parametre som bliver sendt med er *xmlFile* indeholder hvilken XML dokument det skal læses fra, *chartNr* indeholder grafens nummer da skal læses i XML dokumentet, *background* indeholder baggrund farven på siden.
- *Når makeChart.jsp* har fået data fra XML dokumentet kalder det så på metoden *createChart*, med parameterne *data(String[][]), screenWidt(int), screenHeight(int)*, i *Chart.java*.
- *createChart* metoden vil så returnere en *BufferImage* af *JFreechart* som bliver så skrevet ud på ”response outputstream” som en ”png” billede.
- Efter det vil *index.jsp* kunne hente billedet og vise grafen på brugergrænsefladen.

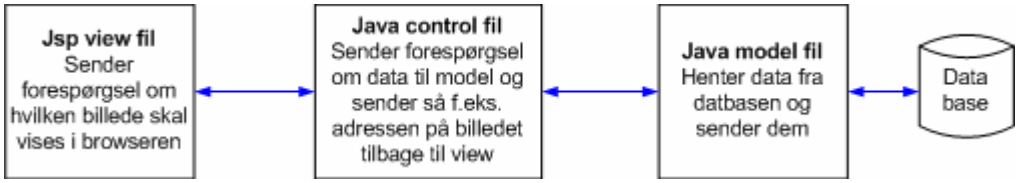
Nu skal jeg lige forklar hvad da ske i metoden *createChart*:

1 Metoden får data ind som definerer hvorhen data skal hentes fra database.
2 Så det første som sker, er at data bliver hentet fra databasen sætte i en *TimeSerie(minuter, data)*. Data for hver enkelt linie bliver så tilføjet i et *TimeSeriesCollection* dataset(som er data for hele grafen).
3 Derefter bliver graferne dannet. Grafen er lavet ved hjælp af et *JFreeChart* bibliotek, som jeg har importeret i min applikation. Dette gratis Java bibliotek gør det nemmer for udvikler at vise professional grafer i deres applikationer
4 Metoden returnerer en Bufferimage af grafen.

5.3 Dynamikken i koden

Applikationen har jeg prøvet at implementere så dynamisk som muligt, f.eks. når browseren er åben og viser en graf ses at tiden ændrer sig og derfor er der ikke bruge for hele tiden at trykke på *refresh*.

Det med at opdatere web browserens indehold har jeg lavet på den måde at sætte et meta tag i head i min html kode <*meta http-equiv="refresh" content="300"*>. Det som det gør, er at sige at denne side skal refresh hver 300 sek eller hver 5 min.



Figur 35

Til at *index.jsp* kan få lamperne til at ændre farve, vise hvilken fase det kører, finde hvilken målere er aktiver, finde alle Journal osv. bruger jeg Java kode i mine jsp filer. Jeg har implementeret tre hoved klasser, *StatusForProcess.java, StatusForSensors.java og StatusForJournal.java*. Alle klasserne er controller, i mit MVC model. De har alle sammen metoder kalder på metoder i Model med parametre med oplysninger om hvorhen data ligger som de vil få og modtager data fra modellen igen. Ud fra de data som de modtager tilbage, finder metoderne ud hvad der skal sendes til view.

Metode	**Returnere**
getErrorStatus(String table)	String
getWatchdog(String tables)	String
getPhaseNumber(String table)	String
getPhasePicture(String table)	String
getATS_State(String table)	String

Tavle 1: Metoder implementeret i *StatusForProcess.java*

- Alle metoder i tavlen ovenpå undtaget *getPhaseNumber*, finder ud af hvilket billede det skal vises på boksene i brugergrænsefladen. Det gør de ved at beregne ud komponentens tilstand fra data som kommer fra Model delen, dvs. fra databasen.

- Metoderne undtaget *getPhaseNumber* returnerer en String da indeholder adressen på billedet som skal vises på brugergrænsefladen.
- Metoden *getPhaseNumber* returnerer nummeret på fasen som kører i øjeblikket på anlægget. Metoden returnerer String med fase nummeret.

Metode	**Returnere**
getJournalStatus()	List<Journal>

Tavle 2: Metoden implementeres i *StatusForJournal.java*

- getJournalStatus metoden henter alle Journaler som findes i databasen. Den bruger hjælp klassen *Journal.java*, som indeholder oplysninger om hver Journal.
- *Journal.java* gemmer oplysningerne id, lastReply, message, replyCount, subject, timestamp, og user.
- Metoden returnerer så List af Journaler.

Metode	**Returnere**
getSensorsStatus()	List<Sensors>

Tavle 3: Metoden implementeres i *StatusForSensors.java*

- getSensorsStatus metoden henter alle sensor som bruges i systemet. Den bruger en hjælp klasse *Sensors.java* som indeholder oplysninger om hver enkelte Sensor.
- *Sensors.java* gemmer oplysningerne id, lastChange og value. Metoden returnerer en List af Sensors. Se Bilag C for metode beskrivelse

5.4 XML dokumenterne

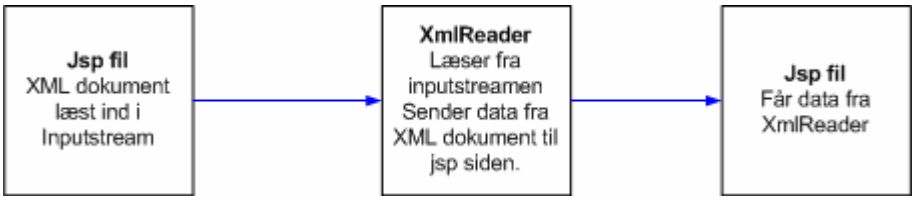
Jeg har brugt XML dokumenter til at indlæse ind bruger informationer og adresse til data. I Star control bruges XML dokumenter til at definere hvilken komponenter det skal vises for de forskellige anlæg. De samme XML dokumenter har jeg så brugt for min mobile udgave af Star. Jeg har sat en attribut (mobile=true/false) i dokumenterne som skal bestemme om den komponent som programmet er i gang med at læse vil være interessant for mobilen eller ej. Jeg har

valgt at bruge de eksisterende XML- dokumenter ellers vil det give double arbejde i fremtiden ved vedligeholdelse når det bliver til et produkt.

Det er tre typer af XML dokumenter som jeg har indlæst for systemet.

1. Det første XML dokument indeholder oplysninger om komponenterne (graf og boks), og hvilken tavle i database der skal hente data fra.
    - **for bokserne** er defineret hvilken type det er, hvorhen den skal linke til hvis det er interessant og hvilken label der skal stå i brugergrænsefladet.
    - **for graferne** er defineret label for Y og X akserne, hvilken farve det skal være på linierne og titel for hver enkelt linie
2. Et andet XML dokument indeholder kun navn på de måler som skal vises for brugeren så det vil være nemmere for brugeren at genkende måleren.
3. Det sidste dokument indeholder hvor mange processer komponenter anlægget har og et link til proces typen side, dvs. hvor mange links det er i undermenuen på **Proces** siden

Når applikationen skal læse et XML dokument starter den at indlæse på en Inputstream på ServletContext, f.eks. *getServletContext().getResourceAsStream("/WEB-INF/xml/plants/brande/home/sensors.xml")*.



Figur 36

Et XML dokument bliver så læst fra Inputstream med hjælp af DOM(Document Object Model) analysere(parser). Jeg har implementeret en klasse som hedder *XmlReader* der indeholder nogle metoder som indlæser de forskellige typer af XML dokumenter. F.eks. hvor mange komponenter(graf eller boks) mobilen skal bruge og finde data for komponenterne.

Metode	Returnere
xmlReadProcessMenu(InputStream inputXML)	String[][]
sensorsRead(InputStream inputXML, String id)	String
xmlReadChart(InputStream inputXML, String chartNumber)	String[][]
getMobileChartNumber(InputStream inputXML)	List<String>
readStatusBoxes(InputStream inputXML, String title)	String[][]
getMobileNumberOfBox(InputStream inputXML)	List<String>

Tavle 4: Metoder implementeres i *XmlReader.java*

- Alle metoderne har en parameter ”Inputstream”. Den definerer hvilket XML dokument metoden skal læse.
- Metoderne *sensorsRead, xmlReadChart* og *readStatusboxes* har også andre parametre, som metoderne bruges til at finde de data som skal returneres.
- Metoden *xmlReadProcessMenu* returnerer Sring array, med to søjle og antal række beregnes ud i koden. Søjle 1 indeholder Display navn for proces og søjle 2 indeholder navnet på xml filen som tilhører for denne proces.
- Metoden *sensorsRead* returnerer String som vil indeholde display navn for Sensor.
- Metoden *xmlReadChart* returnerer String array, med 3 søjle og antal række bliver beregnet ud i koden.
    a. Første rækken og første søjle indeholder teksten for y label på grafen.
    b. Rækkerne bagefter indholder de her værdier i søjlerne
        1 Title for linjen i grafen.
        2 Navnet på tavlen i databasen hvor data skal hentes fra.
        3 Farven på linjen.
- Metoden *readStatusBox* returnerer String array, med 4 søjler og antal række bliver beregnet ud i koden. Her er liste over hvad søjlerne indeholder for hver enkelte række:
        5 Display navn
        6 Navnet på tavlen i databasen hvor dat skal hentes fra.
        7 Hvilket billede type der skal vises.
        8 Hoved titlen for bokset. Er kun gemt i første række.

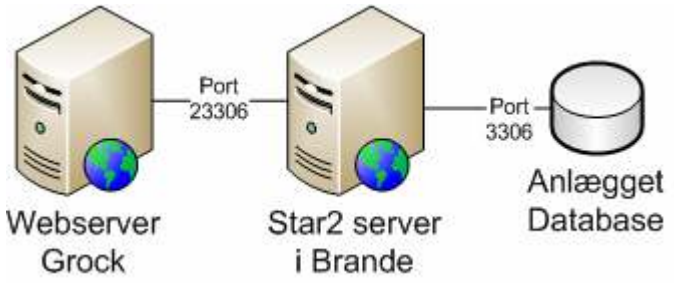
- Metoderne *getMobileChartNumber* og *getMobileNumberOfBox* returnerer List af String som indeholder nummerat på de komponenter som skal vises på brugergrænsefladen.

Se Bilag C for metode beskrivelse.

5.5 Database

Databasen som jeg har adgang til er en *MySQL* database som gemmer alle relevante data for anlægget, både fra målerne og udregnet data. Data bliver skrevet ned i den hver anden minut. Databasen er placeret på renseanlægget Brande og jeg har kun adgang til at hente data der fra, men ingen skrive adgang.

Til at få adgang til en database på anlægget diskutere vi hvordan det vil være bedst at løse det. Inden fra Krügers netværk er adgang til databasen, fordi at en af Krügers IP adresse har adgang der til(se figur 3). En forespørgsel fra en mobile kommer uden fra Krüger med forskellige IP adresser så er det ikke muligt at give adgang til en bestemt IP adresse.



Figur 37

Det blev løst på den måde at åbne en tunnel fra serveren Grock(port 23306), som har adgang til Krügers netværk, til porten(3306) på Star2 serveren i Brande som har så adgang til Brandes database, dvs. at vi har videresendt forespørgsel fra porte 23306 på grock til port 3306 på Brande.

Til at få forbindelse til databasen bruger jeg i min kode JDBC driveren. JDBC står for ”Java Database Connectivity”, som er et API for Java programmering sprog og definerer hvordan en klient skal kommunikere med databasen. API’et har mange metoder som kan bruges i koden til at forespørge og opdatere data. [litt. 9]

Efter at driveren har fået forbindelse til databasen er det muligt i koden at sende SQL statements med hjælp af metoder i jdbc driveren til databasen og få resultat tilbage.

I implementering af applikationen har jeg lavet to klasser som har de roller at skabe forbindelse og sende SQL statement til databasen. De klasser er *DbConnection.java* og *SqlQuery.java*.

Metode	**Returnere**
getConnection()	Connection
getDataset(String columns, String table, TimeSeries series)	TimeSeries
getIntValue(String columns, String table, String columnName)	int
getFloatValue(String columns, String table, String columnName)	float
getSensors()	List<Sensors>
setSensors(String id, float value)	
getJournal()	List<Journal>
createJournal(Journal journal)	
replyJournal(String id, String user, String message)	

Tavle 5: Metoder implementeres i *DbConnection.java og SqlQuery.java*

- Metoden *getConnection* er i *DbConnection.java* og laver forbindelse til databasen.
- Alle metoder undtaget *getConnection* er i *SqlQuery.java*.
- Alle metoder som starter på ”get” er til at hente data fra databasen og få fleste af dem parameter med fra hvilken tavle der skal hentes data fra og nogle har også parameter med fra hvilken søjle det skal hentes fra.
- De ”get” metoder som ikke har nogle parameter er lidt speciale fordi de henter data for **Målere** og **Journal**, og har fast kodet SQL statements.
- Metoderne *setSensors*, *createJournal* og *replyJournal* er alle sammen til at skrive data i databasen. Ligesom jeg nævnte før, så bliver metoderne ikke brugt nu fordi applikationen ikke har skriveadgang til databasen.
- Metoden *getDataset* bruger en hjælp klasse med navnet *Star2Time*. Det som *Star2Time* klassen gør, er at oversætter *Star2Time* til en UTC time eller omvendt. *Star2Time* starter den 1.1.1995 00:00, hvor Star2Time er 0.
- Metoden *getConnection* returnerer forbindelse til databasen som en Connection.
- Metoden *getDataset* returnerer *TimeSeries(minuter, data)* for hver enkelte linje i graferne
- Metoden *getIntValue* returnerer et Integer værdi som skulle findes i databasen.

- Metoden *getFloatValue* returnerer et Float værdi som skulle findes i databasen.
- Metoden *getSensors* returnerer en List af Sensors som findes i systemet.
- Metoden *getJournal* returnerer en List af Journal som findes i systemet

Se Bilag C for metode beskrivelse.

5.6 Skrive ned fra brugergrænsefladet til systemet

Figur 38

I **Journal** og **Målere** fanebladene er det muligt at skrive noget ned til serveren/databasen. Det har jeg løst på den måde at når det er trykket på knappen som sender en aktion med de relevante parametre. Denne aktion sender en forespørgsel til et servlet, hvor metoden *doPost* bliver vækket. *doPost* metoden har den rolle at hente parametrene og sørge for at data bliver sendt til de metoder som skriver data ned i databasen. Efter det sender servlet brugeren videre til **Journal** eller **Målere** faneblade. De her servlet har ikke noget interface.

Som nævnt før er det ikke muligt i det her applikation at skrive ned i databasen. Men koden er skrevet og det mangler kun at fjerne kommentaren fra servlet koden. Det vil også mangle lidt mere for at målerne vil blevet slået fra i systemet, det er at skrive ned i en Nodeserver. Det har jeg valgt ikke at udføre det fordi Krüger har allerede den metode i Star control og det vil være dublering af kode hvis jeg vil skrive en metode som vil kunne det samme.
Se bilag B for metode beskrivelse

5.7 Sikkerhed

Systemet skal være sikkert, dvs. at de som ikke har adgang, ikke kan komme ind. Derfor er det blevet implementeret en ”login” system for det. Her vil jeg forklare opbygningen på login systemet. Jeg har valgt at lave et simple sikkert domain in JBoss, fordi at det vil tage meget længere tid end jeg havde til at lave en kompliceret løsning.

Først vil jeg forklare hvordan man sikrer sit web applikation og bagefter vil jeg forklare hvordan og hvilken log modulen bruges på JBoss.

5.7.1 Web applikation sikkerhed

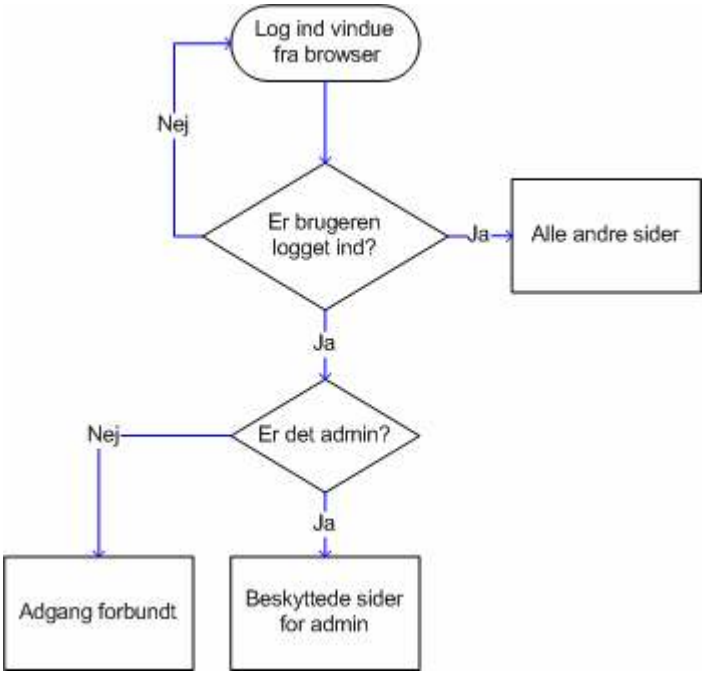
Det gør man med at modificerende *web.xml*(DD for web applikationen) filen for sin applikation. Der er tre tags som bruges til det, aktivere *authentication*, begrænse adgang til specificere URL og definere hvilken bruger typer vil have adgang til applikationen.

5.7.1.1 Aktivere authentication

Web containeren har 4 typer af ” **authentication**” skema: BASIC, DIGEST, FORM og CLIENT-CERT.

- BASIC: *authentication* overfører login information i indkodet form men de er ikke krypteret. BASIC *authentication* er ikke sikker *authentication* protokol fordi brugernavn og password er sendt med simple base64 indkodet(ikke kryptere) som er kendt.
- DIGEST: *authentication* overfører password i krypteret form som giver meget bedre sikkerhed end base64 indkodet som BASIC bruger. Den mekanisme er ikke meget udspredt fordi at JE22 container har ikke kravet om at supporte den.
- CLIENT-CERT: *authentication* overføre login information på et højt sikkerheds form, med bruger ”Public Key Certification(PKC)”.
- FORM *authentication* giver mulighed for at skabe sin egen login form. Login formen skal indeholde felt for brugernavn og password og disse felt skal hedde *j_username* og *j_password*. I *web.xml* filen defineres login form og fejl siden. Den type af *authentication* er mindst sikkert af de fire.

De tre *authentication* mekaniker(BASIC, DIGEST, CLIENT, FORM) ovenpå bruger browserens standard pop-up vindue til at hente brugernavn og password. [litt. 1]



Figur 39

I figur 39 ses opbygningen af sikkerheden for applikationen. Jeg har valgt at bruge BASIC *authentication* til at sikkert min web applikation. Begrundelsen er at den var nem at implementere og den bruger standard pop-up vindue fra browseren til at hente brugernavn og password. Når der indtastes passwordet er bogstaven som er ved at taste ind vist, men de andre er skjulte, men bogstaven bliver skjult når det rigtige bogstav er blevet valgt. Det er vigtigt fordi at hver knap på telefonen har mange bogstaver. I denne her applikation er *authentication* konfiguration i *web.xml* filen defineret på nedenstående måde:

```
<login-config>
      #1<auth-method>BASIC</auth-method>
      #2<realm-name>StarMobile</realm-name>
</login-config>
```

1. *Auth-method* definere hvilken *authentication* bruges i denne web applikation
2. *Realm-name* er kun brugt for at vise formål, mest for web applikation udviklings værktøj.

Hvis man bruger FORM ”*authentication*” skal der definere login side og fejl side. [litt. 10]

5.7.1.2 Begrænse adgang til specificere URL

Til at begrænse adgang til en del af web applikation kan man bruge *security-constraint* element, man kan definere lige så mange som der er brug for. I *security-constraint* element specificeres et eller flere URL "pattern" som brugeren vil have adgang til, det er repræsenteret af hvilken rolle navn brugeren tilhører. Når en bruger som ikke er blevet *authentication* sender forespørgsel til en af de beskyttede URL, vil web containeren åbne login vinduet op og spørge om password og brugernavn.

URL "pattern" kan enten være */journal/journalDialog.jsp* eller *meter/**, hvor * betyder at alle filer i mappen er beskyttet. Når en bruger som er logget ind sender en forespørgsel om et URL får brugeren en side som brugeren ønsker, hvis bruger typen har adgang til den ellers vil brugeren få en fejl side som forklarer at han ikke har adgang til det URL

```
<security-constraint>
      <web-resource-collection>
            <web-resource-name>StarMobile</web-resource-name>
            <url-pattern>/journal/journalDialog.jsp</url-pattern>
            <url-pattern>/journal/journalReply.jsp</url-pattern>
            #1<url-pattern>/meter/*</url-pattern>
      </web-resource-collection>
      <auth-constraint>
            #2<role-name>Admin</role-name>
      </auth-constraint>
       <user-data-constraint>
              <transport-guarantee>NONE</transport-guarantee>
        </user-data-constraint>
 </security-constraint>
```

1. Det tag specificerer hvilken URL har begrænset adgang.
2. Dette tag specificerer hvilken type bruger har adgang til URL som har begrænset adgang.

Når brugeren forespørger URL og ikke har rettigheder til, får han en fejl side. For at undgå at bruger får en standard fejl er det definert i *web.xml* filen hvilken side skal vises ved en bestemt fejl kode, f.eks. 403. Det kan man gøre på følgende måde:

```
<error-page>
      #1<error-code>403</error-code>
      #2<location>/access-denied.jsp</location>
</error-page>
```

1. Definere hvilken fejl kode det er som skal fanges.
2. Hvilken side skal vises når den fejl kode kommer.

[litt. 10]

5.7.1.3 Definere bruger typer adgang til applikationen

Til sidst specificerers alle bruger type navn som har adgang til web applikationen med en *security-role* element for hver rolle. Det gøres på følgende mode:

```
<security-role>
      #1<role-name>Admin</role-name>
  </security-role>
  <security-role>
      <role-name>User</role-name>
  </security-role>
```

- Indeholder navnet på rollen.

[litt. 10]

5.7.2 Godkendelse af bruger i systemet

Til at JBoss kan finde ud af hvilken log modul der skal bruges i log ind procedure, skal der tilføjes en *jboss-web.xml* fil, i WEB-INF katalogen i applikationen. I denne fil skal defineres *jboss-web* element og *security-domain*

```
<jboss-web>
      <security-domain>java:/jaas/StarMobile</security-domain>
</jboss-web>
```

Dette element fortæller JBoss at forbinde web applikation til *StarMobile* sikkerheds domain, som man kan definere i login-config.xml i JBoss. Her har jeg valgt at bruge default domain i JBoss, dvs. hvis domain ikke er defineret er det et standard som bruges. [litt. 17]

Log ind modul som applikationen vil bruge fra JBoss er *UserRolesLoginModule*. *UsersRolesLoginModule* er en simple log ind modul som supporter mange brugere og bruger typer som loades fra Java properties filer. Filen som forbinder brugernavn til password er *user.properties* (bilag I.1) og filen som forbinder brugernavn til bruger type er *role.properties* (bilag I.2). Properties filerne er loadet i installation ved brug af initialiser metoden "thread context class loader". Det betyder at filerne kan placeres ind i J2EE "deployment" JAR, "JBoss" konfiguration katalog, eller i katalog hvor "JBoss" server eller system "classpath" er.
Filen *user.properties* bruger brugernavn=password format for hver bruger i en linje for sig. Det er vist her hvordan jeg har defineret det i min applikation:

```
star=starUser
admin=starUser
```

Filen *role.properties* bruger brugernavn=rolle1, rolle2, format med en valgfri gruppe navn værdi. Her kan man se hvordan jeg har defineret det i min applikation. [litt. 11]

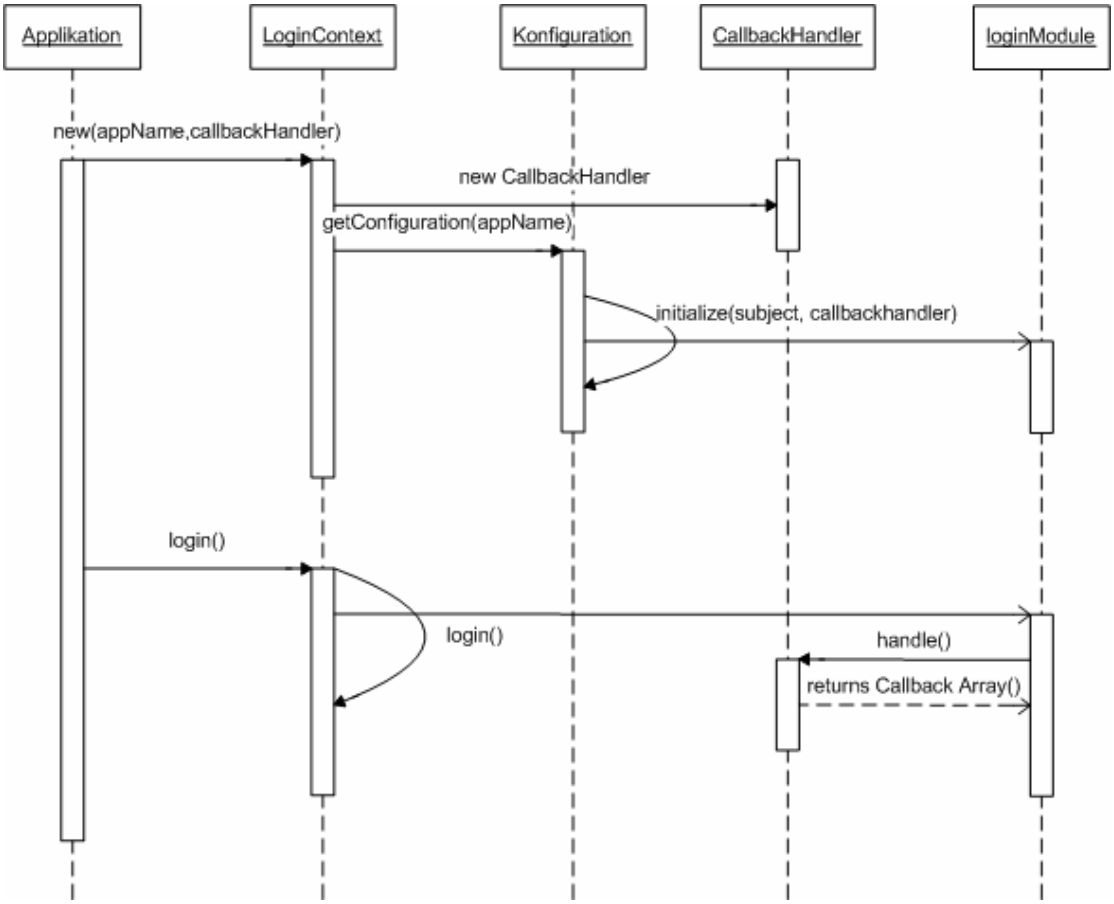
```
star=User
admin=Admin
```

5.7.2.1 JAAS

Log ind modulerne som JBoss har til rådighed for applikationer bruger JAAS(Java Authentication & Authorization Service) Her vil jeg beskrive generelt hvordan en *LoginModule* i JAAS virker.

LoginModule er tilstoppet ind under applikation til at levere en speciel type af *authentication*. De interface som bruges ved implmentering af JAAS er *LoginContext, LoginMobule, Callbackhandler,*

Når Applikationen skriver til *LoginContext* API(Applikation Programmering Interface), forsøger *authentication* teknologen at implementeret *LoginModule* interface. Konfiguration specificeres hvilken *LoginModule* skal bruges af applikationen. Forskellige *LoginModule* kan være klar til opkobling ind i applikationen men uden nogen krav om modifikation til applikation selv.



Figur 40

- *LoginContext* er ansvarlig for at læse en konfiguration fil og starte den rigtige *LoginModule*.
- Hver *LoginModule* er initialiseret med *Subject*, *Callbackhandler*, *delte LoginModule tilstand* og *LoginModule option*.(se figur 40)
- Subjekt repræsenterer bruger eller service som lige har været *authenticatet* og er opdateret af *LoginModule* med relevant *Principale* og *credentials* hvis *authentication* lykkes.
- *LoginModule* bruger *CallbackHandler* til at kommunikere med brugeren f.eks. at få brugernavn og password, som bliver tjekket i *login* metoden som er implementeres i *LoginModule*.
- *CallbackHandler* kan være null, hvis *LoginModule* kræver *authentication subject*, og *Callbackhandler* er null skal *LoginContext* kaste en exception.
- *LoginModules* bruger den delte tilstand til at dele informationer eller data blandt dem selv.

- *LoginModule option* repræsenterer de optioner som konfigurer denne *LoginModule* i log ind konfiguration.
- Optionen er defineret af *LoginModulen* selv og kontroller de opførsler for det. F.eks. *LoginModule* kan definere option til at support debugg/testing evner.
- *LoginModule* gemmer optionen i en Map således at værdien kan hentes hvis man har nøglen til det.
- Der er ingen grænse for antal optioner en *LoginModule* kan vælge at definere.

Applikationen som kalder på en *authentication*, ser den *authentication* processen som en operation med et kald til *LoginContext* login metode. Men alligevel er *authentication* processen i hver *LoginModule* delt i to fase.

1 Den første *authentication* fase er at *LoginContexts* login metode kalder login metode for alle *LoginModule* som er konfigureret. Den login metode i *LoginModule* udfører den aktuel *authentication* og gemmer *authentication* status som privat tilstand. Når den er færdig returnerer *LoginModules* login metoden ”true”(hvis det lykkes) eller ”false” (hvis ikke lykkedes) ellers har den kastet en exception.

2 Den anden fase er at hvis *LoginContext authentcation* lykkes, bliver *commit* metode for hver *LoginModule* kaldt. Den tjekker *LoginModules* gemte privat tilstand til at se om *authentication* lykkedes. Hvis *LoginContext authentication* lykkedes og *LoginModule* egen *authentication* lykkedes, så vil *commit* metoden forbinde de relevante *Principales* og *credentials* (*authentication* data som f.eks. krypton nøgle) med subjekt. Hvis *LoginContext authentication* ikke lykkedes vil abort metode for alle *LoginModule* bliver aktiveret. I dette tilfælde vil *LoginModule* fjerne alle gemt tilstands for den oprindeligt *authentication*.

Når et subjekt skal Logs ud er det kun en fase som bliver involveret. *LoginContext* kalder på *LoginModule* logout metode. Den vil så udføre log ud proceduren, f.eks. fjern Principals eller *credentials* fra subjekt eller session logning informationer. [litt. 12]

6 Test

6.1 Test-Driven Development (TDD)

Test driven development er en teknik hvor man skriver en test case først og så bagefter implementere kode til at bestå testene. Test driven development giver hurtigt feedback til udvikleren.

6.1.1 TDD arbejdsgangen

Her vil jeg forklare lidt arbejdsgangen for en udvikler som bruger TDD

1. **Tilføj test:** Med TDD starter hver ny feature med at skrive testen for den. Testen består selvfølgelig ikke, fordi at kode for featuren er ikke blevet skrevet endnu. Udvikleren er nød til at forstå kraverne for featuren godt.
2. **Kør alle testene og se at de ny test ikke bestå:** Det bekræfter at testene arbejder rigtigt og at testene ikke bestå uden at tilføje nogen ny kode. Dette step tester testen selv i en negativ situation.
3. **Skrive koden:** Det næste skridt er at skrive kode til at få testene til at bestå. Koden som er skrevet på dette her tidspunkt vil ikke være perfekt. Det er vigtigt at koden er skrevet kun til at bestå disse test, men ikke med nogen ekstra virkning.
4. **Køre automatiske test og se om de bestå:** Hvis alle tester bestå kan udvikleren være overbevist om at kode dækker alle testede kraver.
5. **Refractor koden:** Nu er det tid til at rense til i koden hvor det er brug for det. Med at køre testen igen hver gang man har lavet ”refractoring” kan udvikleren være sikker på at ødelægge ikke funktionalitet. Det er vigtigt at fjerne enhver dublering af koden.

Arbejdsgangen er nu gentaget, med start af ny test til at få mere funktionalitet. Størrelsen på trinene kan være så lille som udvikleren ønsker eller så stor som udvikleren føler sig trygt med det. Hvis koden som er skrevet ikke snart opfylder testen kan være at trinet er for stort og derfor kan det være en løsning at skære de ned i nogle mindre bider.

TDD er svært at bruge i nogle tilfælde, som ved implementering af brugergrænsefladet, hvor der skal være en forbindelse til en database, eller hvor systemet har kompliceret input og output. [litt. 13]

Jeg har brugt TDD i min udvikling hvor det var muligt og det har været uden problemer at bruge det. TDD er dog kun brugt ved implementering af to klasser: XmlReader og Star2Time.

6.1.2 Implementerede test i applikationen

XmlReaderTest har fire metoder som tester XmlReader klassen og funktionen. De testmetoder tester om at der er blevet indlæst en fil og at den returnerer de rigtige værdier. De test metoder som jeg har implementeret er:

Metode	**Hvad bliver testet**	**Antal tester**
testRead()	Tester om metoden returnerer noget og om det er de rigtige værdier	3
testSensors()	Tester om metoden returnerer noget og at det er den rigtige label	2
testXmlReadChart()	Tester om metoden returnerer noget og det er de rigtige data. Den tester også antal af grafer som skal vises på interfacet	8
testXmlStatusBox()	Tester om metoden returnerer noget og det er de rigtige data. Den tester også antal af bokse som skal vises på interfacet	10

Alle de her tester bestod, så er det klart at koden arbejder som det er regnet med. (Se kode i bilag J.2)

Star2TimeTest tester klassen Star2Time og funktionen i den. Den tester om omregning på tidspunkt stemmer. Der er implementeret en metode i test klassen:

Metode	**Hvad bliver testet**	**Antal tester**
testUtcToId ()	Tester om at metoden returnerer den rigtige tidspunkt, når den skal omregne UTC tid til Star tid og omvent.	6

Alle testerne i metoden bestod så at klassen Star2Time virker som den er defineret. (Se kode i bilag J.1)

6.2 Test af GUI med black box test

I testen på GUI vil der bruges to typer af bruger. Den ene er en almindelig bruger type som kun har rettighed til at browse gennem siderne(starBruger). Den anden er bruger type (adminBruger) som har nogle ekstra rettighed, bl.a. at skrive noget i systemet.

Hvis det er beskrevet om bruger i testene så er det begge bruger typer som kan lave den aktion.

Test nr.	Tekst	Forventet resultat	Faktisk resultat
1	Bruger logger ind med forkert password	Få log ind vinduet op igen	Som forventet
2	Bruger logger ind med forkert brugernavn	Få log ind vinduet op igen	Som forventet
3	Bruger logger ind med brugernavn og password	Få start siden op med data linket aktiveret	Som forventet
4	Bruger klikker på graf linket på **Forside**n	Graferne vises som skal vise på **Forside**	Som forventet
5	Bruger vælger **Proces** linket i hoved menu	**Proces** side åbnes med Fase aktiveret i under-menu og der under vises Fase graf	Som forventet
6	Bruger vælger data linket under Fase	Data linket under Fase bliver aktiveret	Som forventet
7	Bruger er under Fase-Data, vælger Metal i undermenu under **Proces**	Metal linket under **Proces** bliver aktiv med Data siden aktiveret	Som forventet
8	Bruger vælger graf under Metal og vælger så Returslam	ReturSlam linket under **Proces** bliver aktiv med Graf side aktiveret	Som forventet
9	Bruger vælger **Målere** linket i hoved menuen	Side vises med alle definere måler	Som forventet
10	Bruger vælger **Journal** i hoved menu	Siden vises med en tavle med alle journaler som skrevet har været i systemet	Som forventet
11	Bruger vælger hvilken journal han skal læse, linker på overskrifterne	Side vises med den valgte Journal	Som forventet
12	Bruger trykker på tilbage knappen	**Journal** siden vises igen	Som forventet

13	Almindelig bruger er logget ind og er under **Journal** i hoved menu, trykker på ny besked	Du har ikke adgang til denne side	Som forventet
14	Almindelig bruger vælger journal til at læse	Beskeden vises	Som forventet
15	Almindelig bruger vælger at svare besked	Du har ikke adgang til denne side	Som forventet
16	Almindelig bruger vælger **Målere** i hoved menu og der under vælger han den måler som han vil slå til/fra	Du har ikke adgang til denne side	Som forventet
17	Admin bruger vælger **Målere** i hoved menu og der under han den måler som han vil slå til/fra	Får en side op som spørg han om han vil slå måleren til/fra	Som forventet
18	Admin bruger vælger **Journal** i hoved menu og trykker der under på knappen ny besked	Side vises hvor brugeren kan skrive overskrift og ny besked	Som forventet
19	Admin bruger skriver nogen besked og overskrift, trykker på knappen Reset	Alt det som admin brugeren havde skrevet, bliver fjernet	Som forventet
20	Admin bruger skriver besked og overskrift, der efter trykker på Opret	Beskeden bliver oprettet og **Journal** siden vises	Som forventet
21	Admin bruger vælger besked til at læse	Beskeden bliver vist	Som forventet
22	Admin bruger trykkker på knappen Svar	Side vises hvor brugeren kan skrive ind svaret	Som forventet
23	Admin bruger skriver besked og trykker på Reset	Indtastede beskeder fjernes	Som forventet
24	Admin bruger skriver besked og trykker på Opret	Beskeden sendes og **Journal** siden vises	Som forventet
25	Admin bruger er ind i Ny besked eller svar besked og trykker på Tilbage	**Journal** siden vises	Som forventet
26	Bruger vælger **Forside** og graf	To grafer skulle vises på siden(fra XML fil)	Som forventet

27	Bruger vælger **Forside** og data	4 boks vises på siden (fra XML fil)	Som forventet
28	Brugeren er på data siden	I fase boksen vises fase billedet og fase nummer	Som forventet
29	Brugeren er på data siden	I vagthund boksen vises lamper for vagthund og tekst	Som forventet
30	Brugeren er på data siden	I Data kvalitet boksen vises billeder om kvalitet og tekst om hvilken data det er	Som forventet
31	Brugeren er på data siden	I data boksen for ATS vises billede for ATS og tekst.	Som forventet
32	Bruger vælger **Proces**, Metal og så Data	Ingen boks vises(fra XML fil)	Som forventet
33	Bruger vælger graf under data	Fire grafer vises på siden	Som forventet

6.3 Test konklusion

Ud fra de tester som jeg har lavet, kan jeg sige at systemet fungerer tilfredsstillende. Det mangler dog en funktion i systemet og det er Log ud, det er fordi at når der brugers BASIC *authentication* har den ikke defineret en log ud metode. Når brugeren er logget ind vil bruger informationerne være gemt i browserens cache indtil browseren er lukket ned. Jeg har prøvet at få applikationen til at lukke browseren ned men på mobiltelefonen bliver browseren ikke lukket ned, brugeren får vist adresse linen og derfor bliver bruger informationerne stadigt gemt i cache.

7 Udvidelse af StarMobil

Her kommer nogle punkter som vil være smart at udvide for applikationen. Bagefter vil der forklares hvordan en af dem kan udvikles.

- Udvidelses af StarMobile kunne være at serveren sender en SMS til de bruger da har op givet sit telefonnummer. SMS'en kan indeholde et link til en hjemmeside som vil indeholde oplysninger om udestående alarm. Linket i SMS'en vil identificere brugeren for serveren så brugeren kan få adgang til hjemmesiden. Linket kun være meget langt, og indeholde et kodeord for serveren til at identificere brugeren.
- Lave tooltip for brugeren på brugergrænsefladen. Hvis brugeren ikke husker hvad f.eks. en rød lampe betyder, kunne brugeren nemt se det i tooltip.
- Der skal laves en brugergrænseflade som giver brugeren den mulighed for at indtaste de mobil telefonnummer som serveren skal sende SMS til, brugeren skal selvfølgelig have rettigheder til at tilføje ny nummer.
- Der skal laves en brugergrænseflade for at tilføje ny bruger til StarMobile eller få det til at bruge samme login modul StarControl.
- Hvis projektet bliver til en produkt skal der findes en bedre måde til at hente data på, så det ikke tager så lang tid.
- Brug https i steden for http, ved transport af data over internetet. Det giver mere sikkerhed ved transportering af data.
- En anden udvidelses mulighed for systemet er at ændre *authentcation* metoden, bruge noget som er sikkere en Basic og definere sin egen login domain i login-config.xml filen i JBoss. Prøve også at bruge en anden Loginmodul en UserRolesLoginModule, jeg vil prøve at bruge DatabaseServerLoginModule.

Her kommer vejledninger om hvordan sidste punkten her oven på om DatabaseServerLoginModule kan implementeres.

First vil jeg starte på at ændre jboss-web.xml filen, der skal stå "java:/jaas/ [navn på elementet i application-policy]" i security-domain taget. Når DatabaseServerLoginModule bruges der også lave en fil som hedder database-ds.xml som også skal ligge i WEB-INF katalogen. I denne fil definerer man hvilken

Database der skal forbindes til, hvilken driver det skal bruges, brugernavn og password til databasen, og JNDI-navn(hvis det ikke er defineret er default navn brugt som er ”DefaultDS”). Her neden under ses en opbygning af database-ds.xml:

```
<datasources>
      <local-tx-datasource>
            <jndi-name>StarMobileDS</jndi-name>
             <connection-url>
             jdbc:mysql://localhost:23306/brande</connection-url>
            <driver-class>com.mysql.jdbc.Driver</driver-class>
            <user-name>dbview</user-name>
            <password>readonly</password>
            <metadata>
                  <type-mapping>mySQL</type-mapping>
            </metadata>
      </local-tx-datasource>
</datasources>
```

Efter at den er lavet skal der så tilføjes et tag i login-config.xml filen, hvor der defineres application-policy navn som bruges også i jboss-web.xml, hvilken loginModule der skal bruge, JNDI navn som var defineret i database-ds.xml, og hvor der er to ”SELECT” forespørgsler til databasen som henter password for brugeren og rolle typen som brugeren tilhører.

```
<application-policy name = "StarMobile">
<authentication>
      <login-module code =
"org.jboss.security.auth.spi.DatabaseServerLoginModule"
                                    flag = "required">
             <module-option name =
             "dsJndiName">java:/StarMobileDS</module-option>
             <module-option name = "principalsQuery">SELECT PASSWD
             FROM USERS WHERE USERID=?</module-option>
             <module-option name = "rolesQuery">SELECT USERROLES,
             'Roles' FROM USERROLES WHERE USERID=?</module-option>
      </login-module>
</authentication>
</application-policy>
```

Til sidst skal der huskes at oprette to tavler i databasen, en som hedder USERS og anden som hedder USERROLES. De to tavler vil indeholde det samme som properties filerne som bruges i dette projekt. De kan laves på følgende måde:

```
CREATE TABLE Users(userid VARCHAR(64) PRIMARY KEY, passwd
VARCHAR(64))
CREATE TABLE UserRoles(userid VARCHAR(64), userRoles
VARCHAR(32))
```

8 Konklusion

Jeg har skrevet i 2.4 Valg af løsning at jeg vil prøve så vidt som muligt at bruge Test driven development ved udvikling af det her applikation. Men det gik ikke så godt fordi at jeg ikke kendte metoden til at skrive test for klasser som skulle have adgang til database. Det er også svært at skrive test for GUI og jeg ved ikke hvordan det laves en virtual bruger for en web browser, det var heller ikke den vigtigste del af mit projekt. Jeg har skrevet unit test for de klasser som ikke har brug for adgang til database.

Det er en ting som jeg helt klar vil have implementeret i applikationen, det er LogUd. Log ud var med i designen, men da jeg startede at implementere sikkerheds delen af applikation startede jeg med at bruge FORM *authentication* men fik den ikke til at virke rigtigt. Der var et problem med at når brugeren blev logget inde, kunne serveren ikke fundet ude af hvorhen brugeren skulle sendes næst i applikation og viste derfor hele tiden en fejl meddelelse. Hvis jeg skrev så i adresse linen at der skulle åbnes index.jsp blev siden vist. FORM *authntication* har en log ud metode som BASIC *authentication* ikke har. Den er invalidate( ), hvor session på serveren bliver slettet, den havde jeg også problemer med at få den til at virke.

Når jeg var kommet i gang med at lave adgang for applikation brugte jeg tid på at lave min egen JAAS LoginModule. Jeg fik den til at virke på min egen computer men ikke på serveren. Jeg tror til at få sådan en modul til at virke, er et godt emne i et selvstændigt afgangsprojekt. Det er nemlig mange ting som man skal tænke over når man implementerer et sikkerheds system for en applikation

Jeg har prøvet at tilslutte applikationen på et andet renseanlæg og det gik fint men med en undtagelse, serveren grock kunne ikke finde ud af hvorfra den skulle hente data. Det samme data blev vist på begge applikationerne selvom det var lagt to applikationer med forskellige navn på den.

Jeg har lært meget ved at lave applikationen. Jeg har fået bedre forståelse på hvordan en webapplikation skal opbygge, hvordan en server virker og hvor mange muligheder det er at lave en dynamisk hjemmeside. Med den implementering har jeg vist hvor nemt det kan være at tilføje en lille vigtige feature til et kørende system. Systemet skal bara være designet rigtigt fra starten.

Hvis jeg skulle løse den opgave igen med de erfaringer som jeg har nu, vil jeg selv prøve at opsætte en web server. Ved selv at opsætte web server, vil jeg sikkert

lære langt mere om hvordan en web server virker og de problemer som skal løses. Men det er alt spørgsmål om tid og prioritere af sit arbejde. Det havde så måske ikke været mulighed at hente data fra en live database. Jeg vil også lægge lidt mere vægt på Design delen, fordi nu har jeg mere erfaring af at arbejde med at udvikle noget til en mobil telefon. Ved design af denne her applikation har jeg oplevet hvor svært det kan være at designe noget for en mobil telefon, mest på grund af skærm størrelsen og at web browseren i telefonen er ikke lige så effektive som en almindelig browser.

9 Litteraturliste

[1] Basham, Sierra & Bates, *Head First Servlets & JSP,* O'REILLY, 1st edition, 2004

[2] Tom Marrs & Scott Davis, *JBoss at Work,* O'REILLY, 1st edition, 2005.

[3] *Renseanlæg Damhusåen*, Krüger, 1997 (katalog).

[4] *http://www.aarhuskommune.dk/files/aak/aak/content/filer/magistratens_2._afdeling/aarhus_kommunale_vaerker/spildevand/3.Sxdan_fungerer_et_renseanlxg.pdf*

[5] *Star Superior Tuning and Reporting,* Krüger (katalog)

[6] Forklarer hvad servlet er, *http://da.wikipedia.org/wiki/Java_Servlet*

[7] Forklarer hvad DOM (Document Object Model) er, *http://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20020409/introduction.html*

[8] Forklarer hvordan java kan bruges ved indlæsning på XML dokument http://en.wikipedia.org/wiki/Java_API_for_XML_Processing

[9] Forklarer hvad en JDBC er, *http://da.wikipedia.org/wiki/JDBC*

[10] JAAS(Java Authentication and Authorization Service) bog, *http://www.jaasbook.com/pdfs/jaas-in-action_chapter09-03.pdf*

[11] JBoss bruger håndbog, 8.4.6.2. UsersRolesLoginModule *http://docs.jboss.org/jbossas/jboss4guide/r4/html/ch8.chapter.html*

[12] Introduktion fra Sun om JAAS *http://java.sun.com/javase/6/docs/technotes/guides/security/jaas/JAASLMDevGuide.html#Intro*

[13] Forklarer hvad Test-driven development er, http://en.wikipedia.org/wiki/Test-driven_development

[14] Biblioteket som jeg bruger til at danne grafer, *http://www.jfree.org/jfreechart/*

[15] Forklarer hvordan MVC arkitektur virker og hvorfor man skal bruge MVC i sin udvikling. *http://www.indiawebdevelopers.com/technol1ogy/java/mvcarchitecture.asp*

[16] Java docs, information om Java klasser og hvordan den kan bruges. *http://java.sun.com/j2se/1.5.0/docs/api/*

[17] Forklarer hvordan man kan sikker sit webapplikation *http://wiki.jboss.org/wiki/Wiki.jsp?page=SecureAWebApplicationInJBoss)*

[18] Programmet CSS Tab Designer. *http://www.highdots.com/css-tab-designer/*

[19] Forklarer hvad Apache Ant er. *http://en.wikipedia.org/wiki/Apache_Ant*

[20] Manual for Ant. *http://ant.apache.org/manual/*

[21] Denne side forklarer hvad en war fil er. *http://help.eclipse.org/help31 /index.jsp?topic=/org.eclipse.wst.webtools.doc.user/topics/twcrewar.html*

Bilag A. Tidsplan

Uge nr	1	2	3	4	5	6	7	8	Påske	9	10	
Dato	05/02/2007	12/02/2007	19/02/2007	26/02/2007	05/03/2007	12/03/2007	19/03/2007	26/03/2007	02/04/2007	09/04/2007	16/04/2007	
opsætning/adgang	1											1
design af systemet	3	1	0.5	0.25	0.25	0.25						5
hjemmeside laves		3										3
hjemmesiden viser hvilken tilstand module, med tekst			3									3
hjemmesiden viser logo				1								1
hjemmesiden viser tilstand for modulerne med logo				2								2
hjemmesiden viser graf				1	3							4
hjemmesiden indeholder kun de moduler som anlæget har bruger					1	3						4
brugeren/mobilen får adgang til systemet						1	4		04/04/2007			5
hjemmeside skal kunne sende data til systemet								3		3	1	7
Rapport	1	1	1	1	1	1	1	2		2	4	15
Manddag per uge	5	5	5	5	5	5	5	5		5	5	50

Bilag B. Filer i applikationen

Her kommer list over de filer som jeg har implementeret i min appliktion og hvad de beskæftiger sig med.

Fil navn	**Hvilken rolle den har i systemet**
index.jsp	Den fil finder ud af hvad der skal vises i fanebladene Proces og Forside
makeChart.jsp	Den fil henter graferne fra Chart.java og skriver de ud i en strøm
settings.jsp	Den fil finder hvad der skal vises under fanebladet Målere
access-denied.jsp	Den fil bliver vist når bruger prøver at se side som den har ikke adgang til
journalDialog.jsp	Den fil bliver vist når brugeren vil oprette ny journal
journalReply.jsp	Den fil bliver vist når brugeren vil svar en journal
meterDialog.jsp	Den fil bliver vist når brugeren vil aktivere eller deaktivere en måler
XmlReader.java	Metoderne i denne fil henter data fra XML dokument
DBConnection.java	Den fil laver en forbindelse til en database
SqlQuery.java	Metoderne i denne fil henter data fra database
Star2Time.java	Metoderne i denne fil omregner tid
Chart.java	Metoderne i denne fil laver graferne og skriver dem i en billede
MeterServlet.java	Metoden i denne fil henter data som skal skrives ned i database
JournalServlet.java	Metoden i denne fil henter data som skal skrives ned i database
StatusForProcess.java	Metoderne i denne fil finder ud af hvilken billede skal vises i bokserne
StatusForSensors.java	Metoden i denne fil finder hvilken målere der er i systemet
StatusForJournal.java	Metoden i denne fil finder hvilken Journaler har været skrevet i systemet
Journal.java	Den fil har kun set og get metoder, som indeholder oplysninger om hver enkelte Journal
Sensors.java	Den fil har kun set og get metoder, som indeholder oplysninger om hver enkelt måler
web.xml	Denne fil indeholder oplysninger om applikation for serveren
jboss-web.xml	Denne fil er for serveren for authentication
process.xml	Denne fil indeholder data oplysninger om hvilken processer er i systemet
phase.xml	Denne fil indeholder data oplysninger om phase processen
sensors.xml	Denne fil indeholder display navn for målerne
tabStyle.css	Denne fil definerer udsigten på siden, farve, menu
user.properties	Denne fil indeholder log ind oplysninger, dvs. brugernavn og password
roles.properties	Denne fil indeholder hvilken rolle brugeren tilhører

Bilag C. Metoder i applikationen

Her kommer list over de java metoder som jeg har implementeret i applikationen:

Bilag C.1. StatusForProcess

Metode navn:	getErrorStatus
Input	String table
Output	String
Formål	Det er at finde om det skal vise error billede eller ok billede
Test	Blackbox test
Exception	SQLException

Metode navn:	getWatchdog
Input	String table
Output	String
Formål	Find ud om det skal vise rød, gul eller grøn lampe
Test	Blackbox test
Exception	SQLException

Metode navn:	getPhaseNumber
Input	String table
Output	String
Formål	Finder ud af nummeret på fasen som kører nu
Test	Blackbox test
Exception	SQLException

Metode navn:	getPhasePicture
Input	String table
Output	String
Formål	Finder hvilken fase billede skal vise
Test	Blackbox Test
Exception	SQLException

Metode navn:	getATS()
Input	String table
Output	String
Formål	Finde ud hvilken ATS billede skal vise
Test	Blackbox Test
Exception	SQLException

Bilag C.2. StatusForJournal

Metode navn:	getJournalStatus
Input	Void
Output	List<Journal>
Formål	Finde alle Journal i systemet
Test	Blackbox test
Exception	SQLException

Klassen **Journal** har 7 set og 7 get metoder, for variablerne String id, Timestamp timeStamp, String lastReply, String user, String subject, String message og int replycount.

Bilag C.3. StatusForSensors

Metode navn:	getSensorsStatus
Input	Void
Output	List<Sensors>
Formål	Er at finde liste over alle måler i systemet
Test	Blackbox test
Exception	SQLException

Klassen **Sensors** har 3 get metoder, for variablene String id, String lastChange, int value og 1 set metode for variablen int value.

Bilag C.4. XmlReader

Metode navn:	xmlReadProcessMenu
Input	InputStream inputXML
Output	String[][]
Formål	Finde hvilken processer anlægget har link til deres side
Test	JUnit test
Exception	Exception

Metode navn:	sensorsRead
Input	Inputstream inputXML, String id
Output	String
Formål	Finde display navn for sensor
Test	JUnit test
Exception	Exception

Metode navn:	xmlReadChart
Input	InputStream inputXML, String chartNumber
Output	String[][]
Formål	hente informationer hvor hen data ligger for graf og hvor mange seriers der skal vises
Test	JUnit test
Exception	Exception

Metode navn:	getMobileChartNumber
Input	Inpustream inputXML
Output	List<String>
Formål	Er at finde hvor mange grafer skal vises fra den XML dokument, og lave en liste over nummerer på graferne som skal vises
Test	Junit test
Exception	Exception

Metode navn:	getStatusBoxes
Input	InputStream inputXML, String title

Output	String[][]
Formål	Hente informationer hvor hen data ligger for box som skal vises
Test	JUnit test
Exception	Exception

Metode navn:	getMobileNumberOfBox()
Input	InputStream inputXML
Output	List<String>
Formål	Finde hvor mange boxes skal vises og lave en list over nummer af boxerne som skal vises
Test	JUnit test
Exception	Exception

Bilag C.5. DBConnection

Metode navn:	getConnection
Input	Void
Output	Connection
Formål	Er at lave forbindelse til databasen
Test	Blackbox test
Exception	SQLException

Bilag C.6. SqlQuery

Metode navn:	getDataset
Input	String columns, String table, TimeSeries series
Output	TimeSeries
Formål	At hente data set for graf
Test	Blackbox test
Exception	SQLException

Metode navn:	getIntValue
Input	String columns, String table, String columnName
Output	Int
Formål	Hente value at type int fra database
Test	
Exception	SQLException

Metode navn:	getFloatValue
Input	String columns, String table, String columnName
Output	float
Formål	Hente value at type float fra database
Test	
Exception	SQLException

Metode navn:	getSensors
Input	void
Output	List<Sensors>
Formål	Hente alle sensors i databasen
Test	

Exception	SQLExcetion

Metode navn:	setSensors
Input	String id, float value
Output	void
Formål	Er at ændre status på Sensors i databasen
Test	
Exception	SQLException

Metode navn:	getJournal
Input	void
Output	List<Journal>
Formål	Hente alle Journal fra databasen
Test	
Exception	SQLException

Metode navn:	createJournal
Input	Journal journal
Output	void
Formål	Lave ny journal i databasen
Test	
Exception	SQLExcetion

Metode navn:	replyJournal
Input	String id, String user, String message
Output	void
Formål	Er at tilføj ny beskeder til en journal i databasen
Test	
Exception	SQLException

Bilag C.7. Star2Time

Metode navn:	getUTCtime
Input	int id
Output	long
Formål	Er at oversætte star2 tid i UTC tid
Test	JUnit test
Exception	

Metode navn:	getIDtime
Input	long utcTime
Output	int
Formål	Er at oversætte UTC tid i star2 tid
Test	JUnit test
Exception	

Bilag C.8. Chart

Metode navn:	createChart
Input	String[][] data, int widht, int height

Output	BufferedImage
Formål	Er at samle data for graf og lave en image af grafet
Test	
Exception	IOException, SQLException

Metode navn:	standardChart()
Input	XYDataset defaultXYDataset, String unit, String title
Output	JFreeChart
Formål	Er at lave en graf med mange serier
Test	
Exception	

Bilag C.9. JournalServlet

Metode navn:	doPost
Input	HttpServletRequest request, HttpServletResponse response
Output	void
Formål	Er at hente data fra brugeren og sende data videre.
Test	
Exception	IOException, ServletException

Bilag C.10. MeterServlet

Metode navn:	doPost
Input	HttpServletRequest request, HttpServletResponse response
Output	Void
Formål	Er at hente data fra brugeren og sende data videre.
Test	
Exception	IOException, ServletException

Bilag D. View kode

Bilag D.1. index.jsp

```
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN" "http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta http-equiv="refresh" content="300">
  <title>Star2</title>
  <link rel="STYLESHEET" href="css/tabStyle.css" type="text/css">
  <link rel="STYLESHEET" href="css/box.css" type="text/css">
</head>
<body>
<%@ page import="util.xmlRead.XmlReader"%>
<%@ page import="java.io.*"%>
<%@ page import="status.StatusForProcess" %>
<%@ page import="java.util.List"%>
<%@ page import="java.awt.*"%>
<%@ page import= "javax.imageio.ImageIO"%>
<%--Gets the parameterValues --%>
<%    String[] pageName;
  XmlReader xmlReader = new XmlReader();
  if(request.getParameterValues("pageName") != null){
      pageName = request.getParameterValues("pageName");
  }else{
      pageName = new String[2];
      pageName[0]="index";
  }

  String[] processName ;
  if(request.getParameterValues("processName") != null){
      processName = request.getParameterValues("processName");
  }else{
      processName = new String[2];
```

```
      processName[0]="index";
  }

  String sensorLink;
  String chartLink;
  if(pageName[0].equals("process")){
      sensorLink="/WEB-INF/xml/plants/brande/"+processName[0]+".xml";
      chartLink="xml/plants/brande/"+processName[0]+".xml";
  }else{
      sensorLink="/WEB-INF/xml/plants/brande/home/sensors.xml";
      chartLink="xml/plants/brande/home/charts.xml";
  }
  String[] processType;
  if(request.getParameterValues("processType")!= null){
      processType = request.getParameterValues("processType");
  }else{
      processType= new String[2];
      processType[0]="data";
  }%>
<table width="100%">
  <tr><td>
      <%--The main menu--%>
      <div id="tabsJ">
      <div id="tabline">
            <ul>
                  <%if(pageName[0].equals("process")){%>
                  <li ><a href="index.jsp?pageName=index&amp;processType=data&amp;" ><span>Forside</span></a></li>
                  <li id="current"><a href="index.jsp?pageName=process&amp;processName=process/phase&amp;
                                                            processType=graph&amp;"><span>Proces</span></a></li>
                  <%}else{%>
                  <li id="current"><a href="index.jsp?pageName=index&amp;processType=data&amp;" ><span>Forside</span>
                                                                                        </a></li>
                  <li ><a href="index.jsp?pageName=process&amp;processName=process/phase&amp;processType=graph&amp;">
                                                                        <span>Proces</span></a></li>
                  <%}%>
                  <li ><a href="settings.jsp"><span>Målere</span></a></li>
```

```
                <li ><a href="journal/journal.jsp"><span>Journal</span></a></li>
            </ul>
        </div>
        </div>
</td></tr>
<%if(pageName[0].equals("process")){%>
<tr><td>
        <%--Sub menu, show all processes --%>
        <div id="navPyra">
            <ul >
                <%    ServletContext servletContext = getServletContext();
                InputStream inputStream = servletContext.getResourceAsStream("/WEB-INF/xml/plants/
                                                                        brande/process.xml");
                response.setContentType("text/html; charset=ISO-8859-1");
                String[][] xmlRead = xmlReader.xmlReadProcessMenu(inputStream);
                for(int i = 0; i< xmlRead.length;i++){
                    if(xmlRead[i][1].equals(processName[0])){
                %>
                    <li><a class="active" href="index.jsp?pageName=process&amp;processName=<%=xmlRead[i][1]%>
                                &amp;processType=<%=processType[0]%>&amp;" ><%=xmlRead[i][0]%></a></li>
                <%}else{%>
                    <li><a href="index.jsp?pageName=process&amp;processName=<%=xmlRead[i][1]%>&amp;
                                processType=<%=processType[0]%>&amp;"><%=xmlRead[i][0]%></a></li>
                <%}
                }%>
            </ul>
        </div>
</td></tr>
<%}%>
<tr><td>

        <div id="navSquare">
            <ul>
                <%if(processType[0].equals("graph")){%>
                    <li ><a href="index.jsp?pageName=<%=pageName[0]%>&amp;processName=<%=processName[0]%>&amp;
                                                processType=data&amp;"><span>Data</span></a></li>
```

```
                    <li ><a class="active" href="index.jsp?pageName=<%=pageName[0]%>&amp;processName=
                                    <%=processName[0]%>&amp;processType=graph&amp;"><span>Graf</span></a></li>
               <%}else{%>
                    <li ><a class="active" href="index.jsp?pageName=<%=pageName[0]%>&amp;processName=
                                    <%=processName[0]%>&amp;processType=data&amp;"><span>Data</span></a></li>
                    <li ><a href="index.jsp?pageName=<%=pageName[0]%>&amp;processName=<%=processName[0]%>&amp;
                                                        processType=graph&amp;"><span>Graf</span></a></li>
               <%}%>
          </ul>
     </div>
 </td></tr>
</table>
<%--Data process --%>
<%if(processType[0].equals("data")){%>
 <%--Write out the boxes that will be on the page --%>
 <table   width="320" height="240">
 <tr>
 <%  response.setContentType("text/html; charset=ISO-8859-1");%>
 <%  List<String> mobileNumberOfBox = xmlReader.getMobileNumberOfBox(
                                                        getServletContext().getResourceAsStream(sensorLink));
          for(int j = 0;j<mobileNumberOfBox.size();j++){%>
          <%--Change row --%>
          <%if(j != 0 && j % 2 == 0){%>
                    </tr><tr>
          <% } %>
          <td>
     <%String[][] sensorsData= xmlReader.readStatusBoxes(getServletContext().getResourceAsStream(sensorLink),
                                                                  mobileNumberOfBox.get(j));
          StatusForProcess statusForProcess = new StatusForProcess();%>
               <div class="noprint" id="boxhead" style="width:160px;">
                    <b class="xtop"><b class="xb1"></b><b class="xb2"></b><b class="xb3"></b><b class="xb4">
                                                                                        </b></b>
                    <div class="xboxcontent" i18n:translate=""><%=sensorsData[0][4]%></div>
               </div>
               <div  class="noprint" id="box" style="width:160px;">
                    <div class="xboxcontent" style="padding:4px">
```

```
<div id="trees" style="width:150px;height:100%;overflow:auto">
<table width="150" height="132" align="center">
<%for(int i = 0; i< sensorsData.length;i++){
      //Light box, red,yellow or green
      if(sensorsData[i][3]!=null && sensorsData[i][3].equals("led")) {
            if(sensorsData[i][0]!=null){%>
              <tr>
            <td><a href="<%="process/processGraph.jsp?processName="+sensorsData[i][2]
                        +"&amp;processType=graph&amp"%>"><%=sensorsData[i][0] %></a></td>
              <td><div align="right"><img src="<%=statusForProcess.
            getWatchdog(sensorsData[i][1])%>" width="13" height="13" border="0" /></div></td>
              </tr>
            <%}
            //Status box, ok or error
       }if(sensorsData[i][3].equals("status")) {
                  if(sensorsData[i][0]!=null){%>
                        <tr>
                        <td><a href="<%="process/processGraph.jsp?processName=
      "+sensorsData[i][2]+"&amp;processType=graph&amp"%>"><%=sensorsData[i][0]%></a></td>
                        <td><div align="right"><img src="<%=statusForProcess.
            getErrorStatus(sensorsData[i][1])%>" width="14" height="14" border="0"
                                                                        /></div></td>
                        </tr>
                  <%}%>
            <%--Forecast box --%>
       <%}if(sensorsData[i][3].equals("forecast")) {
                  if(sensorsData[i][0]!=null){%>
                        <tr>
                              <td height="45"><div align="center"><a href="<%=
                              "process/processGraph.jsp?processName="+sensorsData[i][2]+"&amp;
                        processType=graph&amp"%>"><img src="<%=statusForProcess.getATS_State(
                        sensorsData[i][1])%>" width="44" height="28" border="0" />
                                                                  </a></div></td>
                        </tr>
                        <tr>
                              <td height="36"><div align="center">
```

```
                                                                                    <%=sensorsData[i][0]%></div></td>
                                                </tr>
                                          <%}%>
                                     <%--Phase box --%>
                                <%}if(sensorsData[i][3].equals("phase")) {
                                          if(sensorsData[i][0]!=null){%>
                                                <tr>
                                                       <td height="9"><div align="center"><%=sensorsData[0][0]%>:
                                                <%=statusForProcess.getPhaseNumber(sensorsData[0][1])%></div></td>
                                                </tr>
                                                <tr>
                                                      <td height="70"><div align="center"><a href=
                                                "<%="process/processGraph.jsp?processName="+sensorsData[0][2]+"&amp;
                                                      processType=graph&amp"%>"><img src="<%=statusForProcess.
                                                      getPhasePicture(sensorsData[0][1])%>"
                                                       width="60" height="60" border="0" /></a></div></td>
                                                </tr>
                                          <%}%>
                                <%}%>
                          <%}%>
                          </table>
                          </div>
                          </div>
                          <b class="xbottom"><b class="xb4"></b><b class="xb3"></b><b class="xb2"></b><b
                                                                                              class="xb1"></b></b>
                    </div></td>
                    <%if(j==mobileNumberOfBox.size()-1){%>
                          </tr>
                    <%}%>
     <%}%>
</table>
<div align="right">
<table background="images/statusbutton.gif" width="99" height="29">
          <tr>
                <td width="99"><div align="center"><a href="login/logout.jsp">Logout</a></div></td>
          </tr>
```

```
  </table>
<%--Graph process--%>
<%}else{%>
  <%--Gets the screen size  screen.width--%>
  <SCRIPT LANGUAGE="javascript">
      var width = 350
      var height = screen.height
  </SCRIPT>

  <table>
      <%--Write the graphs --%>
      <%    response.setContentType("image/png");
            response.setContentType("text/html; charset=ISO-8859-1");
            String xmlFile=chartLink;//"xml/plants/brande/home/charts.xml";
            List<String> numberOfChart=xmlReader.getMobileChartNumber(getServletContext().getResourceAsStream("/WEB-
                                                                                                    INF/"+xmlFile));
            String chart = null;
            for(int i = 0; i<numberOfChart.size();i++){
            %>
            <tr><td>
                  <SCRIPT LANGUAGE="javascript">
                  document.write("<img src=\"makeChart.jsp? width=100&amp;height=100&amp;screenWidth="+width+"&amp;
                              screenHeight=220&amp;xmlFile=<%=xmlFile%>&amp;chartNr=<%=numberOfChart.get(i)%>&amp;
                              background=ffffff\" width=\""+width+"\" height=\"220\" border=\"1\">")
                  </SCRIPT>
            </td></tr>
      <%}%>
  </table>
<%}%>
</body>
</html>
```

Bilag D.2. makeChart.jsp

```
<%@ page import="util.xmlRead.XmlReader"
%><%@ page import="java.io.*"
```

```
%><%@ page import="java.awt.*"
%><%@ page import= "javax.imageio.ImageIO"
%><%response.setContentType("image/png");
  int screenWidt=Integer.parseInt(request.getParameter("screenWidth"));
  String[] chartNr=request.getParameterValues("chartNr");
  String[] xmlFile = request.getParameterValues("xmlFile");
  XmlReader xmlReader = new XmlReader();
  response.setContentType("text/html; charset=ISO-8859-1");
  //gets chart information where data is stored from XML file
  String[][] data =xmlReader.xmlReadChart(getServletContext().getResourceAsStream("/WEB-
                                                                          INF/"+xmlFile[0]),chartNr[0]);
  OutputStream os = response.getOutputStream();
  //new chart createt
  ImageIO.write(new chartEksample.Chart().createChart(data,screenWidt,220), "png", os);
  os.close();

%>
```

Bilag D.3. settings.jsp

```
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN" "http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><html>
<head>
  <title>Målere</title>
  <link rel="STYLESHEET" href="css/tabStyle.css" type="text/css">
  <link rel="STYLESHEET" href="css/box.css" type="text/css">
  <link rel="STYLESHEET" href="css/list.css" type="text/css">
  <meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=ISO-8859-1">
  <meta http-equiv="refresh" content="300">

</head>

<body>
<table width="100%">
  <tr><td>
       <%--The main menu --%>
       <div id="tabsJ">
```

```
      <div id="tabline">
            <ul>
                  <li ><a href="index.jsp?pageName=index&amp;processType=data&amp;" ><span>Forside</span></a></li>
                  <li ><a href="index.jsp?pageName=process&amp;processName=process/phase&amp;processType=graph&amp;">
                                                                        <span>Proces</span></a></li>
                  <li id="current"><a href="settings.jsp"><span>Målere</span></a></li>
                  <li ><a href="journal/journal.jsp"><span>Journal</span></a></li>
            </ul>
      </div>
      </div>
  </td></tr>
</table>

<%--The table that shows on the page --%>
<div class="noprint" id="boxhead" style="width:100%;">
  <b class="xtop"><b class="xb1"></b><b class="xb2"></b><b class="xb3"></b><b class="xb4"></b></b>
  <div class="xboxcontent" style="text-align:left">
      <table id="theader" class="listTable" width="100%" cellpadding="0" cellspacing="0" >
      <col width="50%">
      <col width="35%">
      <col width="15%">
      <thead>
      <tr>
            <th unselectable="on" >Navn</th>
            <th unselectable="on" >Sidste ændring</th>
            <th unselectable="on" >Status</th>
      </tr>
      </thead>
      </table>
  </div>
</div>
<%@ page import="status.StatusForSensors"%>
<%@ page import="status.Sensors"%>
<%@ page import="util.xmlRead.XmlReader"%>
<%@ page import="java.io.*"%>
<%@ page import="java.util.List"%>
```

```
<%    response.setContentType("text/html; charset=ISO-8859-1"); %>
<%    StatusForSensors statusForSensors = new StatusForSensors();
  response.setContentType("text/html; charset=UTF-8");
  XmlReader xmlReader = new XmlReader();
  List<Sensors> sensorsStatus = statusForSensors.getSensorsStatus(); %>
<div class="noprint" id="box">
                                <%--Calculate rows number --%>
<div class="xboxcontent" style="height:<%=(sensorsStatus.size()*22)+"px"%>;overflow-y:auto" id="tbody">
  <table class="listTable" height="100%" width="100%" cellpadding="0" cellspacing="0"  >
  <col width="50%">
  <col width="35%">
  <col width="15%">
  <tbody>
      <%for(int i = 0; i < sensorsStatus.size();i++){
            Sensors sensors = sensorsStatus.get(i);
            String id= sensors.getId();
            String name = xmlReader.sensorsRead(getServletContext().getResourceAsStream("/WEB-
                                                                  INF/xml/plants/brande/lang/da/sensors.xml"),id);
            %>
            <tr>
                  <td><a color="black" href="meter/meterDialog.jsp?sensorName=<%=name%>&amp;sensorID=<%=id%>&amp;
                                                sensorStatus=<%=sensors.getValue()%>&amp;"><%=name%></a></td>
                  <td><%=sensors.getLastChange()%></td>
                  <td><div align="right">
                        <%
                              if(sensors.getValue() == 1){%>
                                    <img src="images/icons/ok.gif" width="14" height="14" border="0" />ON
                        <%}else{%>
                                    <img src="images/icons/error.gif" width="14" height="14" border="0" />OFF

                        <%}%>
                  </div></td>
            </tr>
      <%}%>
  </tbody>
  </table>
```

```
</div>
  <b class="xbottom"><b class="xb4"></b><b class="xb3"></b><b class="xb2"></b><b class="xb1"></b></b>
</div>
</body>
</html>
```

Bilag D.4. Access-denied.jsp

```
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN" "http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Star Access denied</title>
  <link rel="stylesheet" href="../css/tabStyle.css" type="text/css"/>
</head>

<body>
  <!--The main menu -->
  <div id="tabsJ">
  <div id="tabline">
      <ul>
            <li ><a href="../index.jsp?processType=data&amp;" ><span>Forside</span></a></li>
            <li ><a href="../process/processGraph.jsp?processName=process/phase&amp;
                                                              processType=graph&amp;"><span>Proces</span></a></li>
            <li ><a href="../settings.jsp"><span>Målere</span></a></li>
            <li ><a href="../journal/journal.jsp"><span>Journal</span></a></li>
      </ul>
  </div>
  </div>
  <p>Du har ikke adgang til denne side</p>
  <p><a href="<%= request.getContextPath() %>/index.jsp">Tilbage til forsiden</a>.</p>
</body>
</html>
```

Bilag D.5. journal.jsp

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
  <link rel="STYLESHEET" href="../css/tabStyle.css" type="text/css">
  <link rel="STYLESHEET" href="../css/list.css" type="text/css">
  <link rel="STYLESHEET" href="../css/box.css" type="text/css">
  <title>Journal</title>
</head>

<body>
  <table width="100%">
      <tr><td>
            <div id="tabsJ">
            <div id="tabline">
                  <ul>
                        <li ><a href="../index.jsp?pageName=index&amp;processType=data&amp;" >
                                                                              <span>Forside</span></a></li>
                        <li ><a href="../index.jsp?pageName=process&amp;processName=process/phase&amp;
                                                                    processType=graph&amp;"><span>Proces</span></a></li>
                        <li ><a href="../settings.jsp"><span>Målere</span></a></li>
                        <li id="current"><a href="../journal/journal.jsp"><span>Journal</span></a></li>
                  </ul>
            </div>
            </div>
      </td></tr>
  </table>

<table height="100%" width="100%" cellpadding="0" cellspacing="0">
<%@ page import="status.Journal"%>
<%@ page import="status.StatusForJournal"%>
<%@ page import="java.util.List"%>
<%    String[] journalID=null;
  List<Journal> journalStatus = new StatusForJournal().getJournalStatus();
  if(request.getParameterValues("journalID")== null){%>
      <tr class="noprint">
            <td valign="top" onselectstart="event.returnValue=false;">
```

```
<table>
	<tr>
		<form action="journalDialog.jsp">
			<td ><input class="button1" type="submit" value="Ny besked"/></td>
		</form>
	</tr>
</table>
<div id="boxhead" style="width:100%;">
<b class="xtop"><b class="xb1"></b><b class="xb2"></b><b class="xb3"></b><b class="xb4"></b></b>
<div class="xboxcontent" style="text-align:left">
	<table id="theader" class="listTable" width="100%" cellpadding="0" cellspacing="0">
		<col width="80">
		<col width="150">
		<col width="65">
		<col>
		<thead>
			<tr>
				<th >Sidste svar</th>
				<th >Overskrift</th>
				<th >Svar</th>
				<th >Bruger</th>
			</tr>
		</thead>
	</table>
</div>
</div>
<div id="box">
<div class="xboxcontent" style="height:<%=(journalStatus.size()*22)+"px"%>;overflow-y:auto" id="tbody">
	<table class="listTable" height="0%" width="100%" cellpadding="0" cellspacing="0">
		<col width="80">
		<col width="150">
		<col width="65">
		<col>
		<tbody class="tr.selectedRow">
		<%for(int i =journalStatus.size()-1; i>=0;i--){
			Journal journal= journalStatus.get(i);%>
```

```
                        <tr><td>
                              <%=journal.getLastReply()%>
                        </td>
                        <td>
                              <a href="journal.jsp?journalID=<%=journal.getId()%>&amp;">
                                                                        <%=journal.getSubject()%></a>
                        </td>
                        <td>
                              <%=journal.getReplycount()%>
                        </td>
                        <td>
                              <%=journal.getUser()%>
                        </td></tr>
                  <%}%>
                  </tbody>
            </table>
      </div>
      <b class="xbottom"><b class="xb4"></b><b class="xb3"></b><b class="xb2"></b><b class="xb1"></b></b>
      </div>
  </td>
 </tr>
<%}else{
 journalID = request.getParameterValues("journalID");%>
 <tr>
      <td height="100%" valign="top">
      <%
      int i=0;
      //find the Journal that user want to read in the listen
      while(!journalStatus.get(i).getId().equals(journalID[0])){
                  i++;
      }
      Journal journal = journalStatus.get(i);%>
      <table >
            <tr>
            <form action="journalDialog.jsp">
                  <td ><input class="button1" type="submit" value="Ny besked"/></td>
```

```
            </form>
            <form action="journalReply.jsp">
                  <input type="HIDDEN" name="journalID" value="<%=journal.getId()%>"/>
                  <td ><input class="button1" type="submit" value="Svar"/></td>
            </form>
            </tr>
      </table>
      <%--The box that shows the message --%>
      <div id="boxhead" style="width:100%;">
      <b class="xtop"><b class="xb1"></b><b class="xb2"></b><b class="xb3"></b><b class="xb4"></b></b>
      <div class="xboxcontent" style="text-align:left;padding:4px;height:40px">
            <table id="theader" class="listTable" width="100%" cellpadding="0" cellspacing="0">
                  <thead>
                        <tr><th>Overskrift: <%=journal.getSubject()%></th></tr>
                        <tr><th>Dato: <%=journal.getLastReply()%>       Bruger: <%=journal.getUser()%></th></tr>
                  </thead>
            </table>
      </div>
      </div>
      <div id="box">
      <div class="xboxcontent" style="height:150px";overflow-y:auto" id="tbody">
            <table  height="0%" width="100%" cellpadding="0" cellspacing="0">
                  <tbody>
                        <tr><td>
                              <%= journal.getMessage()%>
                        </td></tr>
                  </tbody>
            </table>
      </div>
            <b class="xbottom"><b class="xb4"></b><b class="xb3"></b><b class="xb2"></b><b class="xb1"></b></b>
      </div>
      </td>
</tr>
<tr>
      <form action="journal.jsp">
         <td ><input class="button1" type="submit" value="Tilbage"/></td>
```

```
      </form>
  </tr>
<%}%>
</table>

</body>
</html>
```

Bilag D.6. journalDialog.jsp

```
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN" "http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <link rel="STYLESHEET" href="../css/tabStyle.css" type="text/css">
  <title>Journal ny besked</title>
</head>

<body>
<table>
  <tr><td>
      <%--The main menu --%>
      <div id="tabsJ">
      <div id="tabline">
            <ul>
                        <li ><a href="../index.jsp?pageName=index&amp;processType=data&amp;" >
                                                                        <span>Forside</span></a></li>
                        <li ><a href="../index.jsp?pageName=process&amp;processName=process/phase&amp;
                                                          processType=graph&amp;"><span>Proces</span></a></li>
                        <li ><a href="../settings.jsp"><span>Målere</span></a></li>
                        <li id="current"><a href="../journal/journal.jsp"><span>Journal</span></a></li>
            </ul>
      </div>
      </div>
  </tr></td>
  <tr><td>
  <%--Send an action to servlet if it receive submit. Servlet will get the data as parameter --%>
  <form method="POST" action="createJournal">
```

```
      <table>
      <tr>
            <td valign="top" width = "40px">Overskrift:</td>
          <td width="200px"><input type="text" name="subject" style="width:100%" /></td>
      </tr>
      <tr>
          <td valign="top" width="40px" >Besked:</td>
          <td width="200px"><textarea name="comments" style="width:100%" rows="8"></textarea></td>
      </tr>
      </table>
  </tr></td>
  <tr><td>
      <table >
            <tr>
              <td ><input class="button1" type="submit" value="Opret" /></td>
              <td ><input class="button1" type="reset" value="Reset"/></td>
  </form>
            <form action="journal.jsp">
               <td ><input class="button1" type="submit" value="Tilbage"/></td>
            </form>
            </tr>
      </table>
  </tr></td>
</table>
</body>
</html>
```

Bilag D.7. journalReply.jsp

```
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN" "http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <link rel="STYLESHEET" href="../css/tabStyle.css" type="text/css">
  <title>Journal svar besked</title>
</head>

<body>
```

```
<%
  //get the parameter
  String[] journalID = request.getParameterValues("journalID");
%>
<table>
  <tr><td>
      <%--The main menu --%>
      <div id="tabsJ">
      <div id="tabline">
            <ul>
                        <li ><a href="../index.jsp?pageName=index&amp;processType=data&amp;" >
                                                                  <span>Forside</span></a></li>
                        <li ><a href="../index.jsp?pageName=process&amp;processName=process/phase&amp;
                                                            processType=graph&amp;"><span>Proces</span></a></li>
                        <li ><a href="../settings.jsp"><span>Målere</span></a></li>
                        <li id="current"><a href="../journal/journal.jsp"><span>Journal</span></a></li>
                  </ul>

      </div>
      </div>
  </tr></td>
  <tr><td>
      <table>
      <%--Send an action to servlet if it receive submit. Servlet will get the data as parameter --%>
      <form method="post" action="createJournal">
      <input type="HIDDEN" name="journalID" value="<%=journalID[0]%>"/>
            <tr>
                  <td valign="top" width="40px" >Besked:</td>
                <td width="200px"><textarea name="comments" style="width:100%" rows="8"></textarea></td>
            </tr>
      </table>
  </tr></td>
  <tr><td>
      <table >
            <tr>
                  <td ><input class="button1" type="submit" value="Opret"/></td>
```

```
                <td ><input class="button1" type="reset" value="Reset"/></td>
      </form>
                <form action="journal.jsp">
                      <td ><input class="button1" type="submit" value="Tilbage"/></td>
                </form>
            </tr>
      </table>
  </tr></td>
</table>
</body>
</html>
```

Bilag D.8. meterDialog.jsp

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
  <title>Målere Dialog</title>
  <link rel="STYLESHEET" href="../css/tabStyle.css" type="text/css">
  <link rel="STYLESHEET" href="../css/box.css" type="text/css">
  <meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=ISO-8859-1">
</head>
<body>
<table>
  <tr><td>
      <%--The main menu --%>
      <div id="tabsJ">
      <div id="tabline">
            <ul>
                  <li ><a href="index.jsp?pageName=index&amp;processType=data&amp;" ><span>Forside</span></a></li>
                  <li ><a href="index.jsp?pageName=process&amp;processName=process/phase&amp;processType=graph&amp;">
                                                                        <span>Proces</span></a></li>
                  <li id="current"><a href="settings.jsp"><span>Målere</span></a></li>
                  <li ><a href="journal/journal.jsp"><span>Journal</span></a></li>
            </ul>
      </div>
      </div>
```

```
<%
    //Sets the char set type and gets the parameter
    response.setContentType("text/html; charset=ISO-8859-1");
    String sensorsName[] =request.getParameterValues("sensorName");
    String sensorsID[] =request.getParameterValues("sensorID");
    int sensorsStatus =Integer.parseInt(request.getParameter("sensorStatus"));
%>
</tr></td>
<tr><td>
    <%--Box that show question to the userS --%>
    <div id="boxhead" style="width:100%;">
    <b class="xtop"><b class="xb1"></b><b class="xb2"></b><b class="xb3"></b><b class="xb4"></b></b>
    <div class="xboxcontent" style="text-align:left;padding:4px;height:20px">
          <table id="theader" class="listTable" width="100%" cellpadding="0" cellspacing="0">
                <thead>
                      <tr><th>Advarsel!</th></tr>
                </thead>
          </table>
    </div>
    </div>
    <div id="box">
    <div class="xboxcontent" style="height:50px";overflow-y:auto" id="tbody">
          <table height="0%" width="100%" cellpadding="0" cellspacing="0">
                <tbody>
                      <tr><td>
                            Er du sikkert på at du vil slå [ <%=sensorsName[0]%> ]
                            <%if(sensorsStatus == 1){%>
                                  <%="fra"%>
                            <%}else{%>
                                  <%="til"%>
                            <%}%>?
                      </td></tr>
                </tbody>
          </table>
    </div>
    <b class="xbottom"><b class="xb4"></b><b class="xb3"></b><b class="xb2"></b><b class="xb1"></b></b>
```

```
      </div>
  </tr></td>
  <tr><td>
      <table>
            <tr>
                  <td background="../images/button.gif" width="79" height="19" align="center"><a href=
            "../meter/sensorStatus?sensorID=<%=sensorsID[0]%>&amp;sensorStatus=<%=sensorsStatus%>&amp" width="80">
                                                                                    OK   </a></td>
                  <td background="../images/button.gif" width="79" height="19" align="center"><a href=
                                                                  "../settings.jsp" width="80">  Afbryd  </a></td>
            </tr>
      </table>
  </tr></td>
</table>
</body>
</html>
```

Bilag D.9. XmlReader.java

```
package util.xmlRead;
import java.io.IOException;
import java.io.InputStream;
import java.util.ArrayList;
import java.util.List;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;
import org.w3c.dom.NamedNodeMap;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;
import org.xml.sax.SAXParseException;
```

```
/**
 * @author elva.Hartmannsdottir
 *
 */
public class XmlReader {


  /**
   * This method reads XML file for submenu with this structure
   * <pages>
   *  <page title="Fase" src="process/phase.xml" />
   * </pages>
   *
   * @param args
   * @return returns a String array with Title and  file name
   * @throws Exception
   */
  public String[][] xmlReadProcessMenu(InputStream inputXML) throws Exception {
            DocumentBuilderFactory factory =
            DocumentBuilderFactory.newInstance();
            Document document;
            String[][] title = null;

        try {
            DocumentBuilder builder =
                  factory.newDocumentBuilder();
            document = builder.parse(inputXML);
            NodeList parents = document.getChildNodes();
            for(int i =0; i<parents.getLength();i++){
                  Node parent = parents.item(i);
                  NodeList childList = parent.getChildNodes();
                  int j = 0;
                  title = new String[childList.getLength()/2][2];
                  for(int k =0; k< childList.getLength();k++){
                        Node child = childList.item(k);
                        if(child.getAttributes() != null){
```

```
                        title[j][0]=(child.getAttributes().getNamedItem("title").getNodeValue());
                        String nodeValue = child.getAttributes().getNamedItem("src").getNodeValue();
                        title[j++][1]=(nodeValue).substring(0,nodeValue.length()-4);
                    }
                }
            }
            return title;
        } catch (SAXParseException spe) {
            throw new Exception(spe);
        } catch (SAXException sxe) {
            throw new Exception(sxe);
        } catch (ParserConfigurationException pce) {
            throw new Exception(pce);
        } catch (IOException ioe) {
            throw new Exception(ioe);
        }
    }

    /**
     *This method read XML file to find display name for sensor
     *with this structure
     * <catalog>
     *   <message>
     *         <id>MsAer1_O2-ok</id>
     *         <string>Ilt LT1 [mg/l]</string>
     *   </message>
     *
     * @param inputXML
     * @param id
     * @return sensorName as String
     * @throws Exception
     */
    public String sensorsRead(InputStream inputXML, String id) throws Exception{
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

        Document document;
```

```
    try {
          DocumentBuilder builder = factory.newDocumentBuilder();
          document = builder.parse(inputXML);

          NodeList parents = document.getChildNodes();
          for(int i =0; i<parents.getLength();i++){
                Node parent = parents.item(i);
                NodeList childList1 = parent.getChildNodes();
                for(int k =0; k< childList1.getLength();k++){
                      Node child = childList1.item(k);
                      NodeList childList2 = child.getChildNodes();
                      for(int j =0; j< childList2.getLength();j++){
                            if(childList2.item(j).getNodeName().equals("id")){
                                  if(childList2.item(j).getTextContent().equals(id)){
                                        return childList2.item(j+2).getTextContent();
                                  }
                            }
                      }
                }
          }
    } catch (SAXParseException spe) {
          throw new Exception(spe);
    } catch (SAXException sxe) {
          throw new Exception(sxe);
    } catch (ParserConfigurationException pce) {
          throw new Exception(pce);
    } catch (IOException ioe) {
          throw new Exception(ioe);
    }
    return id;
}


/**
 * This method finds the number of chart that should be show.
 * @param inputXML
```

```
 * @return number of Chart
 * @throws Exception
 */
public int getNumberOfChart(InputStream inputXML) throws Exception{
    DocumentBuilderFactory factory =
          DocumentBuilderFactory.newInstance();
    int numberOfChart =0;

    try {
          DocumentBuilder builder =factory.newDocumentBuilder();
          Document document = builder.parse(inputXML);
          NodeList parents = document.getChildNodes();
          for(int i =0; i<parents.getLength();i++){
                Node item = parents.item(i);
                if(item.getAttributes() != null && item.getAttributes().getNamedItem("height")!=null){
                      return Integer.parseInt(item.getAttributes().getNamedItem("height").getNodeValue());
                }
          }
          return numberOfChart;
    } catch (SAXParseException spe) {
          throw new Exception(spe);
    } catch (SAXException sxe) {
          throw new Exception(sxe);
    } catch (ParserConfigurationException pce) {
          throw new Exception(pce);
    } catch (IOException ioe) {
          throw new Exception(ioe);
    }
}


/**
 * This method gets  y label and series( title, tables name, line color,) for chart
 * @param inputXML
 * @param chartNumber
```

```
  * @return a string array with first row and column for y lable, the next rows(from 1 to ..) contain series for
chart
  * @throws Exception
  */
 public String[][] xmlReadChart(InputStream inputXML, String chartNumber) throws Exception{
     DocumentBuilderFactory factory =
           DocumentBuilderFactory.newInstance();
     String[][] data = null;

     try {
           DocumentBuilder builder =factory.newDocumentBuilder();
           Document document = builder.parse(inputXML);
           NodeList parents = document.getChildNodes();
           for(int i =0; i<parents.getLength();i++){
                 NodeList childList = parents.item(i).getChildNodes();
                 for(int k =0; k< childList.getLength();k++){
                       Node child = childList.item(k);
                       if(child.getNodeName().equals("item") && child.getAttributes() != null &&
                                   child.getAttributes().getNamedItem("y").getNodeValue().equals(chartNumber)&&
                             child.getAttributes().getNamedItem("mobile").getNodeValue().equals("true")){
                          NodeList childNodes = child.getChildNodes();
                          for(int j=0;j<childNodes.getLength();j++){
                                if(childNodes.item(j).getNodeName().equals("chart") ){
                                      NodeList childNodes2 = childNodes.item(j).getChildNodes();
                                      data = new String[(childNodes2.getLength())+1][3];
                                      int counter =0;
                                      for(int h = 0; h<childNodes2.getLength();h++){
                                            Node child2 = childNodes2.item(h);
                                            final NodeList childNodes3 = child2.getChildNodes();
                                            if(child2.getNodeName().equals("y-axis")){
                                                  for(int y=0;y<childNodes3.getLength();y++){
                                                        if(childNodes3.item(y).getNodeName().equals("label")){
                                                        data[counter++][0]=childNodes3.item(y).getTextContent();
                                                        }
                                                  }
                                            }
```

```
                                        if(child2.getNodeName().equals("series")){
                                              for(int y=0;y<childNodes3.getLength();y++){
                                                    if(childNodes3.item(y).getNodeName().equals("title")){
                                                    data[counter][0]=childNodes3.item(y).getTextContent();
                                                    }
                                                    if(childNodes3.item(y).getNodeName().equals("source")){
                                                    data[counter][1]=childNodes3.item(y).getTextContent();
                                                    }
                                                    if(childNodes3.item(y).getNodeName().equals("color")){
                                                    data[counter++][2]=childNodes3.item(y).getTextContent();
                                                    }
                                              }
                                              ++counter;
                                        }
                                    }
                                }
                            }
                        }
                    }
                }
            return data;
        } catch (SAXParseException spe) {
            throw new Exception(spe);
        } catch (SAXException sxe) {
            throw new Exception(sxe);
        } catch (ParserConfigurationException pce) {
            throw new Exception(pce);
        } catch (IOException ioe) {
            throw new Exception(ioe);
        }
    }

    /**
     * This method returns the number of chart that will be show
     * @param inputXML
     * @return list with the chart number
```

```
 * @throws Exception
 */
public List<String> getMobileChartNumber(InputStream inputXML) throws Exception{
    DocumentBuilderFactory factory =
          DocumentBuilderFactory.newInstance();
    List<String> data = new ArrayList<String>();

    try {
          DocumentBuilder builder =factory.newDocumentBuilder();
          Document document = builder.parse(inputXML);
          NodeList parents = document.getChildNodes();
          for(int i =0; i<parents.getLength();i++){
                NodeList childList = parents.item(i).getChildNodes();
                for(int k =0; k< childList.getLength();k++){
                      Node child = childList.item(k);
                      if(child.getNodeName().equals("item") && child.getAttributes() != null &&
                                  !child.getAttributes().getNamedItem("y").getNodeValue().equals(null)&&
                                  child.getAttributes().getNamedItem("mobile").getNodeValue().equals("true")){
                            NodeList childNodes = child.getChildNodes();
                            for(int j=0;j<childNodes.getLength();j++){
                                  if(childNodes.item(j).getNodeName().equals("chart") ){
                                        data.add(child.getAttributes().getNamedItem("y").getNodeValue());
                                  }
                            }

                      }
                }
          }
          return data;
    } catch (SAXParseException spe) {
          throw new Exception(spe);
    } catch (SAXException sxe) {
          throw new Exception(sxe);
    } catch (ParserConfigurationException pce) {
          throw new Exception(pce);
    } catch (IOException ioe) {
```

```
            throw new Exception(ioe);
        }
    }



    /**
     * This methode return a double array of {label, src ,view , title(in [0][4])}
     * @param inputXML
     * @param title
     * @return double array of {label, src ,view , title(in [0][4])}
     * @throws Exception
     */
    public String[][] readStatusBoxes(InputStream inputXML, String title) throws Exception{
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

        try {
              DocumentBuilder builder =factory.newDocumentBuilder();
              Document document = builder.parse(inputXML);
              NodeList parents = document.getChildNodes();
              String[][] data=null;
              for(int i = 0; i<parents.getLength();i++){
                    NodeList childList = parents.item(i).getChildNodes();
                    for(int k =0; k< childList.getLength();k++){
                          Node child = childList.item(k);
                          if(child.getNodeName().equals("item")){
                                NodeList childList1 = child.getChildNodes();
                                for(int j=0;j<childList1.getLength();j++){
                                      Node child2 = childList1.item(j);
                                      if(child2.getNodeName().equals("box") && child2.getAttributes()!= null &&
                                            child2.getAttributes().getNamedItem("title").getNodeValue().equals(title)){
                                            NodeList childList2 = child2.getChildNodes();
                                            int counter = 0;
                                            data = new String[childList2.getLength()/2][5];
                                            data[0][4]=child2.getAttributes().getNamedItem("title").getNodeValue();
```

```
                                        for(int t=0; t<childList2.getLength();t++){
                                              Node child3 = childList2.item(t);
                                              NamedNodeMap attributes = child3.getAttributes();
                                              if(attributes != null){
                                                    if(attributes.getNamedItem("link") != null){
                                                    data[counter][2]=attributes.getNamedItem("link").getNodeValue();
                                                    }
                                                    data[counter][3]=attributes.getNamedItem("view").getNodeValue();
                                                    data[counter][0]=attributes.getNamedItem("label").getNodeValue();
                                                    data[counter++][1]=attributes.getNamedItem("src").getNodeValue();

                                              }
                                        }
                                        return data;
                                    }
                                }
                            }
                        }
                    } catch (SAXParseException spe) {
                        throw new Exception(spe);
                    } catch (SAXException sxe) {
                        throw new Exception(sxe);
                    } catch (ParserConfigurationException pce) {
                        throw new Exception(pce);
                    } catch (IOException ioe) {
                        throw new Exception(ioe);
                    }
                    return null;
                }

                /**
                 * This method get the number of chart that should be showed
                 * @param inputXML
                 * @return return the number of the chart that should be showed
                 * @throws Exception
```

```
 */
public List<String> getMobileNumberOfBox(InputStream inputXML) throws Exception{
    DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
    try {
          DocumentBuilder builder =factory.newDocumentBuilder();
          Document document = builder.parse(inputXML);
          NodeList parents = document.getChildNodes();
          List<String> data = new ArrayList<String>();
          for(int i = 0; i<parents.getLength();i++){
                NodeList childList = parents.item(i).getChildNodes();
                for(int k =0; k< childList.getLength();k++){
                      Node child = childList.item(k);
                      if(child.getNodeName().equals("item") && child.getAttributes() != null &&
                                  child.getAttributes().getNamedItem("mobile").getNodeValue().equals("true")){
                            NodeList childList1 = child.getChildNodes();
                            for(int j=0;j<childList1.getLength();j++){
                                  Node child2 = childList1.item(j);
                                  if(child2.getNodeName().equals("box") && child2.getAttributes()!= null &&
                                                !child2.getAttributes().getNamedItem("title").equals(null)) {
                                        data.add(child2.getAttributes().getNamedItem("title").getNodeValue());

                                  }
                            }
                      }
                }
          }
          return data;
    } catch (SAXParseException spe) {
          throw new Exception(spe);
    } catch (SAXException sxe) {
          throw new Exception(sxe);
    } catch (ParserConfigurationException pce) {
          throw new Exception(pce);
    } catch (IOException ioe) {
          throw new Exception(ioe);
    }
```

```
    }
}
```

Bilag E. Model kode

Bilag E.1. DBConnection.java

```
package util.db;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DbConnection {

 private static final DbConnection INSTANCE = new DbConnection();

 static public DbConnection getInstance(){
     return INSTANCE;
 }


 /**
  * This methode returns the database connection
  * @return
  * @throws SQLException
  */
 public Connection getConnection() throws SQLException {
     Connection connection= DriverManager.getConnection("jdbc:mysql://localhost:23306/brande", "dbview",
"readonly");
     return connection;
 }

}
```

Bilag E.2. SqlQuery.java

```
package util.db;
```

```
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.Timestamp;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;

import org.jfree.data.time.Minute;
import org.jfree.data.time.TimeSeries;

import status.Journal;
import status.Sensors;
import util.star2Time.Star2Time;

public class SqlQuery {
  private Star2Time star2Time;
  private SimpleDateFormat simpleDateFormat;

  public SqlQuery(){
      star2Time = new Star2Time();
      simpleDateFormat = new SimpleDateFormat("dd-MM-yyyy HH:mm");
  }

  /**
   * This method gets the dataset from database for graph
   * @param columns
   * @param table
   * @param series
   * @return TimeSeries dataset
   * @throws SQLException
   */
  public  TimeSeries getDataset(String columns, String table, TimeSeries series) throws SQLException{
      Connection connection = null;
```

```
    try {
          String sql= "SELECT "+columns
          +" FROM "+table;
          Class.forName("com.mysql.jdbc.Driver");
          connection = DbConnection.getInstance().getConnection();
          Statement stmt = connection.createStatement();
          stmt.setFetchSize(200);
          ResultSet rs;
          rs = stmt.executeQuery(sql);
          while(rs.next()){
                series.add(new Minute(new Date(star2Time.getUTCtime(rs.getInt("id")))), rs.getFloat("value"));
          }
          return series;
    } catch (SQLException e) {
          throw new SQLException("Unable to get data connection");
    } catch (ClassNotFoundException e) {
          throw new SQLException("Unable to find jdbc driver");
    }finally{
          if(connection != null){
                connection.close();
          }
    }
}

/**
 * This method get at value from database
 * @param columns
 * @param table
 * @param columnName
 * @return value as int
 * @throws SQLException
 */
public int getIntValue(String columns, String table, String columnName) throws SQLException{
    Connection connection = null;
    try {
          String sql= "SELECT "+columns
```

```
            +" FROM "+table;
            Class.forName("com.mysql.jdbc.Driver");
            connection = DbConnection.getInstance().getConnection();
            Statement stmt = connection.createStatement();
            stmt.setFetchSize(20);
            ResultSet rs;
            rs = stmt.executeQuery(sql);
            if(rs.next()){
                  return rs.getInt(columnName);
            }
            return 0;
      } catch (SQLException e) {
            throw new SQLException("Unable to get data connection?");
      } catch (ClassNotFoundException e) {
            throw new SQLException("Unable to find jdbc driver");
      }finally{
            if(connection != null){
                  connection.close();
            }
      }
}

/**
 * This method gets a float value from database
 * @param columns
 * @param table
 * @param columnName
 * @return value as float
 * @throws SQLException
 */
public float getFloatValue(String columns, String table, String columnName) throws SQLException{
      Connection connection = null;
      try {
            String sql= "SELECT "+columns
            +" FROM "+table;
            Class.forName("com.mysql.jdbc.Driver");
```

```
            connection = DbConnection.getInstance().getConnection();
            Statement stmt = connection.createStatement();
            stmt.setFetchSize(20);
            ResultSet rs;
            rs = stmt.executeQuery(sql);
            if(rs.next()){
                  return rs.getFloat(columnName);
            }
            return 0;
      } catch (SQLException e) {
            throw new SQLException("Unable to get data connection");
      } catch (ClassNotFoundException e) {
            throw new SQLException("Unable to find jdbc driver");
      }finally{
            if(connection != null){
                  connection.close();
            }
      }
}

/**
 * This method get all sensors that uses
 * @return list of snsors
 * @throws SQLException
 */
public List<Sensors> getSensors() throws SQLException{
      Connection connection = null;
      List<Sensors> sensors = new ArrayList<Sensors>();
      String sql="SELECT S.name, S.value, T.tid FROM `settings_message` S, ("+
      "SELECT max( sid ) AS sid, name FROM `settings_message` "+
      "WHERE name LIKE \"%-ok\" GROUP BY name )Temp, `settings_thread` T "+
      "WHERE Temp.name = S.name AND Temp.sid = S.sid AND Temp.sid = T.sid";
      try {
            Class.forName("com.mysql.jdbc.Driver");
            connection = DbConnection.getInstance().getConnection();
            Statement stmt = connection.createStatement();
```

```
            stmt.setFetchSize(200);
            ResultSet rs;
            rs = stmt.executeQuery(sql);
            while(rs.next()){
                  String id = rs.getString(1);
                  int value = (int)rs.getFloat(2);
                  Timestamp timestamp = rs.getTimestamp(3);
                  sensors.add(new Sensors(id, simpleDateFormat.format(timestamp),value));
            }

      } catch (SQLException e) {
            throw new SQLException("Unable to get data connection");
      } catch (ClassNotFoundException e) {
            throw new SQLException("Unable to find jdbc driver");
      }finally{
            if(connection != null){
                  connection.close();
            }
      }
      return sensors;
}
/**
 * This method setSensors value
 * @param id
 * @param value
 * @throws SQLException
 */
public void setSensors(String id, float value) throws SQLException{
      Connection connection= null;
      try {
            String title = id + ": ";
            if (value == 0.0) {
                  title += "ON -> OFF";
            } else {
                  title += "OFF -> ON";
            }
```

```
            String sql= "INSERT INTO `settings_thread`(`tid`,`title`)\n"+
            "VALUE (`0`, `"+title+"`)";

            Class.forName("com.mysql.jdbc.Driver");
            connection = DbConnection.getInstance().getConnection();
            Statement stmt = connection.createStatement();
            stmt.setFetchSize(200);
            stmt.executeUpdate(sql);
            ResultSet generatedKeys = stmt.getGeneratedKeys();
            int tid=0;
            if(generatedKeys.next()){
                  tid=generatedKeys.getInt(1);
            }
            sql= "INSERT INTO `settings_message`(`sid`,`name`,`value`)"+
            "VALUE (`"+tid+"`,`"+id+"`,`"+value+"`)";
      } catch (SQLException e) {
            throw new SQLException("Unable to get data connection");
      } catch (ClassNotFoundException e) {
            throw new SQLException("Unable to find jdbc driver");
      }finally{
            if(connection != null){
                  connection.close();
            }
      }

}
/**
 * This method get all Journal
 * @return a list of Journal
 * @throws SQLException
 */
public List<Journal> getJournal() throws SQLException{
      Connection connection = null;
      List<Journal> journal = new ArrayList<Journal>();
      String sql="SELECT t.tid AS id, date AS timestamp, lastReply, user, subject, message, replyCount " +
      "FROM  journal_threads AS t, journal_replies AS r " +
```

```
      "WHERE  t.tid = r.tid GROUP BY  t.tid";
      try {
            Class.forName("com.mysql.jdbc.Driver");
            connection = DbConnection.getInstance().getConnection();
            Statement stmt = connection.createStatement();
            stmt.setFetchSize(200);
            ResultSet rs;
            rs = stmt.executeQuery(sql);
            while(rs.next()){
                  journal.add(new Journal(rs.getString("id"),
rs.getTimestamp("timestamp"),simpleDateFormat.format(rs.getTimestamp("lastReply")),rs.getString("user"),
                              rs.getString("subject"),rs.getString("message"),rs.getInt("replyCount")));
            }

      } catch (SQLException e) {
            throw new SQLException("Unable to get data connection");
      } catch (ClassNotFoundException e) {
            throw new SQLException("Unable to find jdbc driver");
      }finally{
            if(connection != null){
                  connection.close();
            }
      }
      return journal;
  }

  /**
   * This method creates a new Journal in database
   * @param journal
   * @throws SQLException
   */
  public void createJournal(Journal journal ) throws SQLException{
      Connection connection = null;
      int time=0;
      if(journal.getUser() != null && journal.getSubject() != null && journal.getMessage() != null){
            try {
```

```
                String sql= "INSERT INTO `journal_threads`(`tid`,`subject`,`replyCount`,`lastReply`)\n"+
                "VALUE (0,`"+journal.getSubject()+"`,`0`, NOW())";

                Class.forName("com.mysql.jdbc.Driver");
                connection = DbConnection.getInstance().getConnection();
                Statement stmt = connection.createStatement();
                stmt.setFetchSize(200);
                stmt.executeUpdate(sql);
                ResultSet generatedKeys = stmt.getGeneratedKeys();
                if(generatedKeys.next()){
                      time=generatedKeys.getInt(1);
                }
                sql= "INSERT INTO `journal_replies`(`rid`,`tid`,`date`,`user`,`message`)"+
                "VALUE (0,`"+time+"`,NOW(),`"+journal.getUser()+"`,`"+journal.getMessage()+"`)";
          } catch (SQLException e) {
                throw new SQLException("Unable to get data connection");
          } catch (ClassNotFoundException e) {
                throw new SQLException("Unable to find jdbc driver");
          }finally{
                if(connection != null){
                      connection.close();
                }
          }
     }
}

/**
 * This method writes a new message to a old Journal
 * @param id
 * @param user
 * @param message
 * @throws SQLException
 */
public void replyJournal(String id, String user, String message) throws SQLException{
     Connection connection = null;
     if(id != null && user != null && message != null){
```

```
            try {
                  String sql= "INSERT INTO `journal_replies`(`rid`,`tid`,`date`,`user`,`message`)\n"+
                  "VALUE (0,`"+id+"`,NOW(),`"+user+"`,`"+message+"`)";

                  Class.forName("com.mysql.jdbc.Driver");
                  connection = DbConnection.getInstance().getConnection();
                  Statement stmt = connection.createStatement();
                  stmt.setFetchSize(200);
                  stmt.executeUpdate(sql);
                  sql="UPDATE `journal_threads` SET `replyCount`= `replyCount`+1, `lastReply` = NOW()\n"+
                        "WHERE `tid` ="+id+";";
                  stmt.executeUpdate(sql);
            } catch (SQLException e) {
                  throw new SQLException("Unable to get data connection");
            } catch (ClassNotFoundException e) {
                  throw new SQLException("Unable to find jdbc driver");
            }finally{
                  if(connection != null){
                        connection.close();
                  }
            }
      }
  }
}
```

Bilag E.3. Star2Time.java

```
package util.star2Time;

import java.util.GregorianCalendar;
import java.util.TimeZone;

public class Star2Time {
  long star2Delta =120000L; // in milliseconds
  long star2Time = 0L;
```

```
 public Star2Time(){
     GregorianCalendar gregorianCalendar = new GregorianCalendar(2005,0,1,0,0,0);
     gregorianCalendar.setTimeZone(TimeZone.getTimeZone("GMT+0"));
     star2Time=gregorianCalendar.getTimeInMillis();
 }

 /**
  * This method translate starTime to UTCTime
  * @param idTime
  * @return UTCtime as long
  */
 public long getUTCtime(int idTime) {
     return ((long)(idTime-1))*star2Delta+star2Time;
 }

 /**
  * This method translate UTCtime to starTime
  * @param utcTime
  * @return starTime as int
  */
 public int getIDtime(long utcTime) {
     return (int) ((utcTime-star2Time+star2Delta/2)/star2Delta)+1;
 }

}
```

Bilag F. Controller kode

Bilag F.1. Chart.java

```
package chartEksample;

import java.awt.Color;
import java.awt.image.BufferedImage;
import java.io.IOException;
import java.sql.SQLException;
import java.text.DecimalFormat;
import java.text.NumberFormat;

import org.jfree.chart.ChartFactory;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.axis.NumberAxis;
import org.jfree.chart.plot.XYPlot;
import org.jfree.chart.renderer.xy.XYItemRenderer;
import org.jfree.data.time.Minute;
import org.jfree.data.time.TimeSeries;
import org.jfree.data.time.TimeSeriesCollection;
import org.jfree.data.xy.XYDataset;

import util.db.SqlQuery;

public class Chart {
  private SqlQuery sqlQuery;
  private DecimalFormat decimalFormat;

  public Chart(){
      sqlQuery = new SqlQuery();
      decimalFormat = (DecimalFormat)NumberFormat.getNumberInstance();
      decimalFormat.applyPattern("#.#");
  }
```

```
 /**
  * This methode made a bufferedImage af chart
  * @param data
  * @param widht
  * @param height
  * @return a bufferedImage
  * @throws IOException
  * @throws SQLException
  */
 public BufferedImage createChart(String[][] data,int widht, int height ) throws IOException, SQLException{
     TimeSeriesCollection dataset = new TimeSeriesCollection();
     int maxId=0;
     int min =0;
     try{
           maxId = sqlQuery.getIntValue("MAX(`id`)", "`"+data[1][1]+"`", "MAX(`id`)");
           min=maxId-180;
           for(int i = 1; i<data.length;i++){
                 if(data[i][0]!= null)
                       dataset.addSeries(sqlQuery.getDataset("`id` , `value`", "`"+data[i][1]+"` WHERE `id`
<"+maxId+" AND `id` >"+min, new TimeSeries(data[i][0],Minute.class)));
           }
     }catch (SQLException e) {
           throw new SQLException("Unable to get data for chart");
     }
     //create chart
     JFreeChart chart = standardChart(dataset,data[0][0],"");
     //create BufferedImage of chart
     return chart.createBufferedImage(widht, height);
 }

 /**
  * This method create a standard chart
  * @param defaultXYDataset
  * @param unit
  * @param title
  * @return a JFreeChart
```

```
   */
  private JFreeChart standardChart(XYDataset defaultXYDataset, String unit, String title) {
      JFreeChart chart;
      chart = ChartFactory.createTimeSeriesChart(title, "", unit, defaultXYDataset, true, false,false);
      chart.setBackgroundPaint(Color.white);

      XYPlot plot = (XYPlot) chart.getPlot();
      plot.setBackgroundPaint(Color.white);
      plot.setDomainGridlinePaint(Color.lightGray);
      plot.setRangeGridlinePaint(Color.lightGray);

      XYItemRenderer valueRenderer = chart.getXYPlot().getRenderer();
      valueRenderer.setSeriesPaint(0, Color.BLUE);
      valueRenderer.setSeriesPaint(1, Color.RED);
      valueRenderer.setSeriesPaint(2, Color.GREEN);
      valueRenderer.setSeriesPaint(3, Color.BLACK);
      chart.getXYPlot().setRenderer(valueRenderer);

      final NumberAxis rangeAxis = (NumberAxis) plot.getRangeAxis();
      rangeAxis.setNumberFormatOverride(decimalFormat);
      return chart;
  }



}
```

Bilag F.2. MeterServlet.java

```
package servlet;

import java.io.IOException;
import java.sql.SQLException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import util.db.SqlQuery;

public class MeterServlet extends HttpServlet {

  /**
   *
   */
  private static final long serialVersionUID = 1L;

  /* this method sends data til database and sends further to settings.jsp
   * @see javax.servlet.http.HttpServlet#doPost(javax.servlet.http.HttpServletRequest,
javax.servlet.http.HttpServletResponse)
   */
  public void doPost(HttpServletRequest request,
            HttpServletResponse response)
  throws IOException, ServletException {
      SqlQuery sqlQuery = new SqlQuery();
//          try {
            if(request.getParameterValues("sensorID") != null
                        && request.getParameterValues("sensorStatus") != null){
                  String[] sensorsID = request.getParameterValues("sensorID");
                  float sensorsStatus = Float.parseFloat(request.getParameter("sensorStatus"));
//                      sqlQuery.setSensors(sensorsID[0], sensorsStatus);
            }else{
                  response.sendRedirect("meterDialog.jsp?sensorName=FEJL &amp;sensorID= &amp;sensorStatus= &amp;");
            }
//          } catch (SQLException e) {
//                throw new IOException(e.toString());
//          }
      response.sendRedirect("../settings.jsp");
  }

  public void doGet(HttpServletRequest request,
```

```
            HttpServletResponse response)
  throws IOException, ServletException {
      doPost(request, response);
  }

}
```

Bilag F.3. JournalServlet.java

```
package servlet;

import java.io.IOException;
import java.sql.SQLException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import status.Journal;
import util.db.SqlQuery;

public class JournalServlet extends HttpServlet {


  /**
   *
   */
  private static final long serialVersionUID = 1L;

  /*
   * This methode sendes data to database and sends further to journal.jsp
   * @see javax.servlet.http.HttpServlet#doPost(javax.servlet.http.HttpServletRequest,
javax.servlet.http.HttpServletResponse)
   */
  public void doPost(HttpServletRequest request,
            HttpServletResponse response)
```

```
  throws IOException, ServletException {
      SqlQuery sqlQuery = new SqlQuery();
      String[] comments = request.getParameterValues("comments");
//          try {
            if(request.getParameterValues("subject") != null){
                  String[] subject = request.getParameterValues("subject");
                  //sqlQuery.createJournal(new
Journal(null,null,null,request.getRemoteUser(),subject[0],comments[0],0));
            }else if(request.getParameterValues("journalID")!= null){
                  String[] journalID = request.getParameterValues("journalID");
//                      sqlQuery.replyJournal(journalID[0], request.getRemoteUser(), comments[0]);
            }
//          } catch (SQLException e) {
//                throw new IOException(e.toString());
//          }
      response.sendRedirect("journal.jsp");
  }


}
```

Bilag F.4. StatusForProcess.java

```
package status;

import java.sql.SQLException;

import util.db.SqlQuery;

public class StatusForProcess {

  private SqlQuery sqlQuery;
  public StatusForProcess(){
      sqlQuery = new SqlQuery();
  }
```

```
/**
 * This method returns the name of picture that should be show for status
 * @param table
 * @return return a path string
 * @throws SQLException
 */
public String getErrorStatus(String table) throws SQLException{
    int maxValue = sqlQuery.getIntValue("MAX(`id`)", table, "MAX(`id`)");
    int value = (int) sqlQuery.getFloatValue("`quality`", table+"  WHERE  `id`="+maxValue, "quality");
    if(value == 0){
          return "images/icons/ok.gif";
    }else{
          return "images/icons/error.gif";
    }
}


/**
 * This method returns the name of picture that should be show for lights
 * @param tables
 * @return return a path string
 * @throws SQLException
 */
public String getWatchdog(String tables) throws SQLException{
    String table1=tables.substring(0, tables.indexOf(","));
    String table2=tables.substring(tables.indexOf(",")+1);
    int maxValue = sqlQuery.getIntValue("MAX(`id`)","`"+ table1+"`" , "MAX(`id`)");
    int light = (int) (sqlQuery.getFloatValue("`value`","`"+ table1+"`  WHERE  `id`="+maxValue, "value")
               +sqlQuery.getFloatValue("`value`","`"+  table2+"`  WHERE  `id`="+maxValue, "value"));
    if(light == 0){
          return "images/icons/led_red.gif";
    }else if(light ==2 || light == 1){
          return "images/icons/led_yellow.gif";
    }else{
          return "images/icons/led_green.gif";
    }
```

```
}

/**
 * This method return the pahse number
 * @param table
 * @return string number
 * @throws SQLException
 */
public String getPhaseNumber(String table) throws SQLException{
    int maxValue = sqlQuery.getIntValue("MAX(`id`)", "`"+table+"`","MAX(`id`)");
    return Integer.toString((int)sqlQuery.getFloatValue("`value`", "`"+table+"` WHERE `id`="+maxValue, "value"));
}

/**
 * This method returns the name of the picture that should be show for phase
 * @param table
 * @return return a path string
 * @throws SQLException
 */
public String getPhasePicture(String table) throws SQLException{
    final String phaseNumber = getPhaseNumber(table);
    return "images/fase/fase_"+phaseNumber.substring(0,3)+".png";
}

/**
 * This method returns the name of the picture that should be show for forecast
 * @param table
 * @return return a path string
 * @throws SQLException
 */
public String getATS_State(String table) throws SQLException{
    int maxValue = sqlQuery.getIntValue("MAX(`id`)", table,"MAX(`id`)");
    int state=((int)sqlQuery.getFloatValue("`value`", table+" WHERE `id`="+maxValue, "value"));
    if(state == 2){
          return "images/icons/forecast_rain.gif";
    }else if(state == 1){
```

```
            return "images/icons/forecast_cloudy.gif";
      }else{
            return "images/icons/forecast_sunny.gif";
      }
  }
}
```

Bilag F.5. StatusForSensors.java

```
package status;

import java.sql.SQLException;
import java.util.List;

import util.db.SqlQuery;

public class StatusForSensors {

  private SqlQuery sqlQuery;
  public StatusForSensors(){
      sqlQuery = new SqlQuery();
  }

  /**
   * This method gets a list of Sensors
   * @return return a list of sensors
   * @throws SQLException
   */
  public List<Sensors> getSensorsStatus() throws SQLException{
      return sqlQuery.getSensors();
  }

}
```

Bilag F.6. StatusForJournal.java

```
package status;

import java.sql.SQLException;
import java.util.List;

import util.db.SqlQuery;

public class StatusForJournal {

 private SqlQuery sqlQuery;
 public StatusForJournal(){
     sqlQuery = new SqlQuery();
 }

 /**
  * This method get Journal and returns a list of Journal
  * @return
  * @throws SQLException
  */
 public List<Journal> getJournalStatus() throws SQLException{
     return sqlQuery.getJournal();
 }


}
```

Bilag F.7. Journal.java

```
package status;

import java.sql.Timestamp;

public class Journal {

 private String id;
 private Timestamp timeStamp;
 private String lastReply;
```

```
private String user;
private String subject;
private String message;
private int replycount;

public Journal(String id, Timestamp timestamp, String lastReply, String user, String subject,
          String message, int replycount) {
    this.id=id;
    this.timeStamp=timestamp;
    this.lastReply= lastReply;
    this.user = user;
    this.subject=subject;
    this.message=message;
    this.replycount = replycount;
}

/**
 * @return the id
 */
public String getId() {
    return id;
}

/**
 * @param id the id to set
 */
public void setId(String id) {

    this.id = id;
}

/**
 * @return the lastReply
 */
public String getLastReply() {
    return lastReply;
```

```
}

/**
 * @param lastReply the lastReply to set
 */
public void setLastReply(String lastReply) {
    this.lastReply = lastReply;
}

/**
 * @return the message
 */
public String getMessage() {
    return message;
}

/**
 * @param message the message to set
 */
public void setMessage(String message) {
    this.message = message;
}

/**
 * @return the replycount
 */
public int getReplycount() {
    return replycount;
}

/**
 * @param replycount the replycount to set
 */
public void setReplycount(int replycount) {
    this.replycount = replycount;
}
```

```
/**
 * @return the subject
 */
public String getSubject() {
    return subject;
}

/**
 * @param subject the subject to set
 */
public void setSubject(String subject) {
    this.subject = subject;
}

/**
 * @return the timeStamp
 */
public Timestamp getTimeStamp() {
    return timeStamp;
}

/**
 * @param timeStamp the timeStamp to set
 */
public void setTimeStamp(Timestamp timeStamp) {
    this.timeStamp = timeStamp;
}

/**
 * @return the user
 */
public String getUser() {
    return user;
}
```

```
  /**
   * @param user the user to set
   */
  public void setUser(String user) {
      this.user = user;
  }

}
```

Bilag F.8. Sensors.java

```
package status;

public class Sensors {

  private String id;
  private String lastChange;
  private int value;

  public Sensors(String id, String lastChange, int value) {
      this.id = id;
      this.lastChange = lastChange;
      this.value = value;
  }

  /**
   * @return the id
   */
  public String getId() {
      return id;
  }

  /**
   * @return the lastChange
   */
  public String getLastChange() {
      return lastChange;
```

```
  }

  /**
   * @return the value
   */
  public int getValue() {
      return value;
  }

  public void setValue(int value){
      this.value = value;
  }


}
```

Bilag G. XML

Bilag G.1. web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">

    <display-name>StarMobile Application</display-name>
    <description>
  </description>


    <servlet>
        <servlet-name>JournalServlet</servlet-name>
        <servlet-class>servlet.JournalServlet</servlet-class>
    </servlet>
     <servlet-mapping>
        <servlet-name>JournalServlet</servlet-name>
        <url-pattern>/journal/createJournal</url-pattern>
    </servlet-mapping>
    <servlet>
        <servlet-name>SensorServlet</servlet-name>
        <servlet-class>servlet.MeterServlet</servlet-class>
    </servlet>
     <servlet-mapping>
        <servlet-name>SensorServlet</servlet-name>
        <url-pattern>/meter/sensorStatus</url-pattern>
    </servlet-mapping>
    <error-page>
      <error-code>403</error-code>
      <location>/access-denied.jsp</location>
    </error-page>
```

```
    <session-config>
      <session-timeout>10</session-timeout>
    </session-config>
  <security-constraint>
      <web-resource-collection>
            <web-resource-name>StarMobile</web-resource-name>
            <url-pattern>/*</url-pattern>
      </web-resource-collection>
      <auth-constraint>
            <role-name>User</role-name>
            <role-name>Admin</role-name>
      </auth-constraint>
       <user-data-constraint>
             <transport-guarantee>NONE</transport-guarantee>
        </user-data-constraint>
  </security-constraint>

<security-constraint>
  <web-resource-collection>
      <web-resource-name>StarMobile Admin</web-resource-name>
      <url-pattern>/journal/journalDialog.jsp</url-pattern>
      <url-pattern>/journal/journalReply.jsp</url-pattern>
      <url-pattern>/meter/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
      <role-name>Admin</role-name>
  </auth-constraint>
   <user-data-constraint>
            <transport-guarantee>NONE</transport-guarantee>
       </user-data-constraint>
</security-constraint>

  <security-role>
      <role-name>User</role-name>
  </security-role>
  <security-role>
```

```
      <role-name>Admin</role-name>
  </security-role>

  <login-config>
      <auth-method>BASIC</auth-method>
      <realm-name>StarMobile Realm</realm-name>
  </login-config>

   </web-app>
```

Bilag G.2. jboss-web.xml

```
<?xml version='1.0' encoding='UTF-8' ?>

<!DOCTYPE jboss-web
     PUBLIC "-//JBoss//DTD Web Application 2.3V2//EN"
     "http://www.jboss.org/j2ee/dtd/jboss-web_3_2.dtd">

<jboss-web>
  <security-domain>java:/jaas/other</security-domain>
  <context-root>/StarMobile</context-root>
</jboss-web>
```

Bilag G.3. build.xml

```
<project name="StarMobileTest" default="all" basedir=".">
  <target name="jarFile">
      <jar destfile="lib\userRoles.jar" basedir="src\webapp\properties"></jar>
  </target>
  <target name="warpackage" depends="jarFile">
      <war destfile="StarMobile.war" webxml="src/webapp/WEB-INF/web.xml">
            <fileset dir="src/webapp/">
                  <exclude name="properties/*"/>
            </fileset>
            <lib dir="lib/jfreechart-1.0.3">
                  <include name="jfreechart-1.0.3.jar"/>
```

```
            </lib>
            <lib dir="lib/jfreechart-1.0.3/lib">
                  <include  name="jcommon-1.0.6.jar"/>
            </lib >
            <lib dir="lib">
                  <include  name="userRoles.jar"/>
            </lib>
            <classes dir="bin">
                  <include  name="chartEksample\Chart.class"/>
                  <include  name="mypackage\Hello.class"/>
                  <include  name="servlet\*"/>
                  <include  name="status\*"/>
                  <include  name="util\db\*"/>
                  <include  name="util\xmlRead\XmlReader.class"/>
                  <include  name="util\star2Time\Star2Time.class"/>
            </classes>
            <zipfileset dir="src"  prefix="WEB-INF/src">
                  <exclude name="webapp"/>
            </zipfileset>
            <zipfileset dir="xml"
                prefix="WEB-INF/xml"/>
            <zipfileset dir="src/webapp/images"
                          prefix="images"/>
      </war>
  </target>
  <target name="all" depends="warpackage" />
</project>
```

Bilag G.4. process.xml

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<pages>
  <page title="Fase" src="process/phase.xml" />
  <page title="Ilt" src="process/oxygen.xml" />
  <page title="Returslam" src="process/returnsludge.xml" />
  <page title="Metal" src="process/metal.xml" />
  <page title="ATS" src="process/ats.xml" />
```

```
</pages>
```

Bilag G.5. phase.xml

Alle andre xml filer for process og forside er bygged op på samme måde som phase.xml. Derfor har jeg valgt kun at vise phase.xml, de andre kan ses på CD

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<grid width="6" height="5">

  <!--Auto Charts Phase -->
  <item x='1' y='1'  width='4' mobile="true"> <chart ><legend/><x-axis type='relative'><delta>360</delta></x-
axis><y-axis><label>[mg/l]</label></y-axis><series type='line'><title>NH4dq
LT1</title><source>DqAer1_Nh4</source><color>blue</color></series><series type='line'><title>NO3dq
LT1</title><source>DqAer1_No3</source><color>red</color></series><series type='line'><title>PO4dq
LT1</title><source>DqAer1_Po4</source><color>green</color></series><series type='line'><title>LT1
udlÃ¸b</title><source>MsAer1_Ukip</source><color>black</color></series></chart></item>

  <item x='1' y='2'  width='4' mobile="true"> <chart ><legend/><x-axis type='relative'><delta>360</delta></x-
axis><y-axis><label>[-]</label></y-axis><series type='line'><title>Fase
LT1</title><source>MsAer1_Phase</source><color>blue</color></series><series type='line'><title>Fase SP
LT1</title><source>PsAer1_PhaseSp</source><color>red</color></series></chart></item>

  <item x='1' y='3'  width='4' mobile="true"> <chart ><legend/><x-axis type='relative'><delta>360</delta></x-
axis><y-axis><label>[mg/l]</label></y-axis><series type='line'><title>NH4pr
LT1</title><source>GeAer1_Nh4Pr</source><color>blue</color></series><series type='line'><title>NH4dq
LT1</title><source>DqAer1_Nh4</source><color>red</color></series><series type='line'><title>NO3pr
LT1</title><source>GeAer1_No3Pr</source><color>green</color></series><series type='line'><title>NO3dq
LT1</title><source>DqAer1_No3</source><color>black</color></series></chart></item>

  <item x='1' y='4'  width='4' mobile="true"> <chart ><legend/><x-axis type='relative'><delta>360</delta></x-
axis><y-axis><label>[mg/l]</label></y-axis><series type='line'><title>O2
LT1</title><source>DqAer1_O2</source><color>blue</color></series><series type='line'><title>O2
LT2</title><source>DqAer2_O2</source><color>red</color></series><series type='line'><title>O2 SP
LT1</title><source>DoAer1_Gui</source><color>green</color></series><series type='line'><title>O2 SP
LT2</title><source>DoAer2_Gui</source><color>black</color></series></chart></item>
```

```
 <item x='1' y='5'  width='4' mobile="true"> <chart ><legend/><x-axis type='relative'><delta>360</delta></x-
axis><y-axis><label>kl</label></y-axis><series type='line'><title>%N-tid
LT1</title><source>RpAer1_%NtidCy</source><color>blue</color></series><series type='line'><title>%N-tid
LT2</title><source>RpAer2_%NtidCy</source><color>red</color></series><series type='line'><title>OUR
[g/m3/h]</title><source>OuAer12_OurCy</source><color>green</color></series><series type='line'><title>Cy
lgd[min]</title><source>Fs_LgdCy</source><color>black</color></series></chart></item>


 <!-- row 1 sensor-->
 <item x="5" y="1" width="1" mobile="true">
     <box title="Vagthund">
          <sensor view="led" type="watchdog" label="Fase" src="Ms_FsWd,Fs_Wd"/>
          <sensor view="led" type="watchdog" label="Ilt" src="Ms_O2Wd,Il_Wd"/>
          <sensor view="led" type="watchdog" label="ATS" src="Ms_AtWd,At_Wd"/>
     </box>
 </item>
 <item x="6" y="1" width="1" mobile="true">
     <box title="Fase">
          <sensor view="phase" label="phase" src="MsAer1_Phase"/>
     </box>
 </item>
 <!-- row 2 sensor-->

 <item x="6" y="2" width="1" mobile="false">
     <box title="Data kvalitet - DQC">
          <sensor view="status" type="quality"  label="NH4 LT1" src="DqAer1_Nh4" link="control/phase"/>
          <sensor view="status" type="quality"  label="NO3 LT1" src="DqAer1_No3" link="control/phase"/>
          <sensor view="status" type="quality"  label="PO4 LT1" src="DqAer1_Po4" link="control/phase"/>
     </box>
 </item>
 <item x="5" y="2" width="1" mobile="false">
     <box title="Styring status">
          <sensor view="status" type="quality"  label="PrimÃ¦rstyring" src="DqPs_FaultState1"
link="control/phase"/>
          <sensor view="status" type="quality"  label="1. Tilbagefald (OUR)" src="DqPs_FaultState2"
link="control/phase"/>
```

```
            <sensor view="status" type="quality"  label="SRO" src="DqPs_FaultStatesro" link="control/phase"/>
      </box>
  </item>
</grid>
```

Bilag G.6. sensors.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<catalog>
  <message>
      <id>MsAer1_O2-ok</id>
      <string>Ilt LT1 [mg/l]</string>
  </message>
  <message>
      <id>MsAer2_O2-ok</id>
      <string>Ilt LT2 [mg/l]</string>
  </message>
  <message>
      <id>DoAer1_O2Sp-ok</id>
      <string>Sætpunkt: Ilt LT1 [mg/l]</string>
  </message>
  <message>
      <id>DoAer2_O2Sp-ok</id>
      <string>Sætpunkt: Ilt LT2 [mg/l]</string>
  </message>
  <message>
      <id>MeAer_QmetalSp-ok</id>
      <string>Sætpunkt: Metaldosering LT [l/h]</string>
  </message>
  <message>
      <id>Ms_AtWd-ok</id>
      <string>ATS vagthund fra PLC [-]</string>
  </message>
  <message>
      <id>Ms_FsWd-ok</id>
```

```
    <string>Fase vagthund fra PLC [-]</string>
</message>
<message>
    <id>Ms_MeWd-ok</id>
    <string>Metaldosering vagthund fra PLC [-]</string>
</message>
<message>
    <id>Ms_O2Wd-ok</id>
    <string>Iltsætpunkt vagthund fra PLC [-]</string>
</message>
<message>
    <id>Ms_RsWd-ok</id>
    <string>Returslam vagthund fra PLC [-]</string>
</message>
<message>
    <id>MsAer_Ssout-ok</id>
    <string>SS ud af LT [g/l]</string>
</message>
<message>
    <id>MsAer_SsoutState-ok</id>
    <string>Fejlsignal: SS ud af LT [-]</string>
</message>
<message>
    <id>MsAer_Temp-ok</id>
    <string>Temperatur LT [C]</string>
</message>
<message>
    <id>MsAer1_Ikip-ok</id>
    <string>Indløbskip LT1 [-]</string>
</message>
<message>
    <id>MsAer2_Ikip-ok</id>
    <string>Indløbskip LT2 [-]</string>
</message>
<message>
    <id>MsAer1_Ukip-ok</id>
```

```
    <string>Udløbskip LT1 [-]</string>
</message>
<message>
    <id>MsAer2_Ukip-ok</id>
    <string>Udløbskip LT2 [-]</string>
</message>
<message>
    <id>MsAer1_Nh4-ok</id>
    <string>NH4 LT1 [mg/l]</string>
</message>
<message>
    <id>MsAer1_No3-ok</id>
    <string>NO3 LT1 [mg/l]</string>
</message>
<message>
    <id>MsAer1_Po4-ok</id>
    <string>PO4 LT1 [mg/l]</string>
</message>
<message>
    <id>MsAer1_Nh4State-ok</id>
    <string>Fejlsignal: NH4 LT1 [-]</string>
</message>
<message>
    <id>MsAer1_No3State-ok</id>
    <string>Fejlsignal: NO3 LT1 [-]</string>
</message>
<message>
    <id>MsAer1_Po4State-ok</id>
    <string>Fejlsignal: PO4 LT1 [-]</string>
</message>
<message>
    <id>MsAer1_Phase-ok</id>
    <string>Aktuel fase LT [-]</string>
</message>
<message>
    <id>MsAer1_Rotor-ok</id>
```

```
    <string>Rotor LT1 [antal]</string>
</message>
<message>
    <id>MsAer2_Rotor-ok</id>
    <string>Rotor LT2 [antal]</string>
</message>
<message>
    <id>MsFs_Ssr-ok</id>
    <string>SS i returslam [g/l]</string>
</message>
<message>
    <id>MsFs_SsrState-ok</id>
    <string>Fejlsignal: SS i returslam [-]</string>
</message>
<message>
    <id>MsFs1_Qras-ok</id>
    <string>Returslam flow ET1 [m3/h]</string>
</message>
<message>
    <id>MsFs1_QrasState-ok</id>
    <string>Drift af returpumper ET1 [-]</string>
</message>
<message>
    <id>MsFs2_Qras-ok</id>
    <string>Returslam flow ET2 [m3/h]</string>
</message>
<message>
    <id>MsFs2_QrasState-ok</id>
    <string>Drift af returpumper ET2 [-]</string>
</message>
<message>
    <id>MsIn_Niv-ok</id>
    <string>Niveau i indløb [cm]</string>
</message>
<message>
    <id>MsIn_Ph-ok</id>
```

```
    <string>pH i indløb [-]</string>
</message>
<message>
    <id>MsOut_Ph-ok</id>
    <string>pH i udløb [-]</string>
</message>
<message>
    <id>MsIn_Q-ok</id>
    <string>Indløbsflow [m3/h]</string>
</message>
<message>
    <id>MsIn_Pumpra1-ok</id>
    <string>Indløbspumpe 1 i drift [-]</string>
</message>
<message>
    <id>MsIn_Pumpra2-ok</id>
    <string>Indløbspumpe 2 i drift [-]</string>
</message>
<message>
    <id>MsIn_Pumpra3-ok</id>
    <string>Indløbspumpe 3 i drift [-]</string>
</message>
<message>
    <id>MsIn_Qmetal-ok</id>
    <string>Metaldosering [l/h]</string>
</message>
<message>
    <id>MsOut_Ss-ok</id>
    <string>Ss i udløb [mg/l]</string>
</message>
<message>
    <id>MsOut_SsState-ok</id>
    <string>Fejlsignal: Ss i udløb [-]</string>
</message>
<message>
    <id>MsRjct_Qpolystate-ok</id>
```

```
      <string>Polymerdosering i drift [-]</string>
  </message>
  <message>
      <id>MsWwtp_Rain-ok</id>
      <string>Nedbør [mm]</string>
  </message>
  <message>
      <id>PsAer1_PhaseSp-ok</id>
      <string>Sætpunkt: Fase [-]</string>
  </message>
  <message>
      <id>RsFs1_QrSp-ok</id>
      <string>Sætpunkt: Returflow ET1 [m3/h]</string>
  </message>
  <message>
      <id>RsFs2_QrSp-ok</id>
      <string>Sætpunkt: Returflow ET2 [m3/h]</string>
  </message>
</catalog>
```

Bilag H. Stylesheet

Bilag H.1. tabStyle.css

```
html{font-family: Verdana, Arial, Helvetica, sans-serif;
   background-color:#003871;}
     body {
       color:white;
    background-color:#003871;
    color:white
     margin:0;
     padding:0;
     font: bold 11px/1.5em Verdana;

}


h2 {
        font: bold 14px Verdana, Arial, Helvetica, sans-serif;
        color: #fff;
        margin: 0px;
        padding: 0px 0px 0px 15px;
}

a { color:white;
    text-decoration:none}

/*- Menu Tabs J--------------------------- */

     #tabsJ {
       float:left;
       width:100%;
       background:#003871;
       font-size:100%;
       line-height:normal;
```

```
  }
#tabsJ ul {
    margin:0;
    padding:10px 10px 0 50px;
    list-style:none;
  }
#tabsJ li {
  display:inline;
  margin:0;
  padding:0;
  }
#tabsJ a {
  float:left;
  background:url("../images/tab/tab_left.gif") no-repeat left top;
  margin:0;
  padding:0 0 0 5px;
  text-decoration:none;
}
#tabsJ a span {
  float:left;
  display:block;
  background:url("../images/tab/tab_right.gif") no-repeat right top;
  padding:5px 15px 4px 6px;
  color:#708491
  }

/* Commented Backslash Hack hides rule from IE5-Mac \*/
#tabsJ a span {float:none;}
/* End IE5-Mac hack */
#tabsJ a:hover span {
  color:#FFF;
  }
#tabsJ a:hover {
  background-image: url("../images/tab/tab_left_on.gif");
  }
#tabsJ a:hover span {
```

```
      background-image: url("../images/tab/tab_right_on.gif");
      }
    #tabsJ #current a {
       background-image: url("../images/tab/tab_left_on.gif");
      border-bottom:1px solid #003871;
    }
    #tabsJ #current a span {
              background-image: url("../images/tab/tab_right_on.gif");
              border-bottom:1px solid #003871;
        color:#FFF;
        }
 #tabsJ #tabline{
    background: url("../images/tab/tab_line.gif") repeat-x left bottom;
    line-height: normal;
    height: 34px;
    width: 100%;
}

/* pyramid */

#navPyra {
        margin: 0;
        padding: 0 0 20px 10px;
 }

#navPyra li {
        margin: 0;
        padding: 0;
        display: inline;
        list-style-type: none;
}

#navPyra a:link, #navPyra a:visited {
        float: left;
        font-size: 10px;
        line-height: 10px;
```

```
        font-weight: bold;
        padding: 0 12px 6px 12px;
        text-decoration: none;
        color: #708491;
}

#navPyra a:link.active, #navPyra a:visited.active, #navPyra a:hover {
        color: #FFF;
        background: url(../images/Pyramid.gif) no-repeat bottom center;
}

/* square */

#navSquare {
        margin: 0;
        padding: 0 0 20px 10px;

}

#navSquare li {
        margin: 0;
        padding: 0;
        display: inline;
        list-style-type: none;
}

#navSquare a:link, #navSquare a:visited {
        float: left;
        font-size: 10px;
        line-height: 14px;
        font-weight: bold;
        padding: 0 12px 6px 12px;
        text-decoration: none;
        color: #708491;
}
```

```
#navSquare a:link.active, #navSquare a:visited.active, #navSquare a:hover {
        color: #FFF;
        background: url(../images/Square.gif) no-repeat bottom center;
}

/* BUTTONS */
#button input{
 border:none;
 color: white;
 background-color:transparent;
 background-image:url(../images/button.gif);
 width:80px;
 height:20px;
 padding-bottom: 1px;
 font-size: 11px;

 cursor: pointer;
}

input.button1 {
 border:2px;
 border-color:#FFF;
 color: white;
 background-color:#003871;
 width:80px;
 height:20px;
 padding-bottom: 1px;
 font-size: 11px;

 cursor: pointer;
}
```

Bilag I. Properties files

Bilag I.1. user.properties

```
# The username to password mapping properties file
star=starUser
admin=starUser
```

Bilag I.2. roles.properties

```
# The username to role(s) mapping properties file
star=User
admin=Admin
```

Bilag J. Test

Bilag J.1. Star2TimeTest

```
package util.star2Time;

import java.util.GregorianCalendar;
import java.util.TimeZone;

import junit.framework.TestCase;

public class Star2TimeTest extends TestCase{

  public void testUtcToId(){

      Star2Time star2Time = new Star2Time();
      GregorianCalendar gregorianCalendar = new GregorianCalendar(2005,0,1,0,0,0);
      gregorianCalendar.setTimeZone(TimeZone.getTimeZone("GMT+0"));
      assertEquals("1-The UTCtime should be the same",gregorianCalendar.getTimeInMillis() ,star2Time.getUTCtime(1));
      assertEquals("1-The IDtime should be the same", 1, star2Time.getIDtime(gregorianCalendar.getTimeInMillis()));

      gregorianCalendar.set(GregorianCalendar.MINUTE,2);
      System.out.println((gregorianCalendar.getTimeInMillis()-star2Time.getUTCtime(2)));
      assertEquals("2-The UTCtime should be the same", gregorianCalendar.getTimeInMillis(),star2Time.getUTCtime(2));
      assertEquals("2-The IDtime should be the same", 2, star2Time.getIDtime(gregorianCalendar.getTimeInMillis()));

      gregorianCalendar.set(GregorianCalendar.MINUTE,18);
      System.out.println((gregorianCalendar.getTimeInMillis()-star2Time.getUTCtime(10)));
      assertEquals("3-The UTCtime should be the same",
gregorianCalendar.getTimeInMillis(),star2Time.getUTCtime(10));
      assertEquals("3-The IDtime should be the same", 10, star2Time.getIDtime(gregorianCalendar.getTimeInMillis()));
      GregorianCalendar gCal= new GregorianCalendar();
      gCal.setTimeInMillis(star2Time.getUTCtime(557606));
      System.out.println(gCal.getTime());
```

```
  }

}
```

Bilag J.2. XmlReaderTest

```
package util.xmlRead;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.util.List;


import junit.framework.TestCase;

public class XmlReaderTest extends TestCase {

  public void testRead() throws Exception{
       XmlReader xmlReader = new XmlReader();
       final String[][] xmlRead = xmlReader.xmlReadProcessMenu(new FileInputStream(new
File("xml/plants/brande/process.xml")));
       assertNotNull("string should not be null", xmlRead);
       assertEquals("1.String should be the same","Fase", xmlRead[0][0]);
       assertEquals("2.String should be the same","process/oxygen", xmlRead[1][1]);
       for(int i = 0; i<xmlRead.length;i++){
            System.out.println(xmlRead[i][0]+"  "+xmlRead[i][1]);
       }
  }

  public void testSensors() throws Exception {
       XmlReader xmlReader = new XmlReader();
       File file = new File("xml/plants/brande/lang/da/sensors.xml");
       String xmlString = xmlReader.sensorsRead(new FileInputStream(file),"MsAer2_O2-ok");
```

```
    assertNotNull("String should not be null", xmlString);
    assertEquals("The string should be the same","Ilt LT2 [mg/l]", xmlString);
}

public void testXmlReadChart() throws FileNotFoundException, Exception {
    XmlReader xmlReader = new XmlReader();
    File inputXML = new File("xml/plants/brande/home/charts.xml");
    System.out.println("Chart1");
    String[][] xmlReadChart = xmlReader.xmlReadChart(new FileInputStream(inputXML), "2");
    assertNull("String array should be null",xmlReadChart);

    String[][] xmlReadChart1 = xmlReader.xmlReadChart(new FileInputStream(inputXML), "1");
    assertNotNull("String array should not be null",xmlReadChart1);
    assertEquals("Should be the same legend","[mg/l]", xmlReadChart1[0][0]);
    assertEquals("Should be the same name","NH4 LT1", xmlReadChart1[1][0]);
    assertEquals("Should be the same table name","DqAer1_No3", xmlReadChart1[3][1]);
    assertEquals("Should be the same color","green", xmlReadChart1[5][2]);

    int numberOfChart = xmlReader.getNumberOfChart(new FileInputStream(inputXML));
    assertEquals("Number should be the same",4, numberOfChart);
    inputXML = new File("xml/plants/brande/process/phase.xml");
    List<String> mobileChartNumber = xmlReader.getMobileChartNumber(new FileInputStream(inputXML));
    assertEquals("Number of chart should be the same",5, mobileChartNumber.size());
    assertEquals("Value should be","1", mobileChartNumber.get(0));
}

public void testXmlStatusBox() throws FileNotFoundException, Exception{
    XmlReader xmlReader = new XmlReader();
    File inputXML = new File("xml/plants/brande/home/sensors.xml");
    String[][] readStatusBoxes = xmlReader.readStatusBoxes(new FileInputStream(inputXML), "Vagthund");
    assertNotNull("Should not be null", readStatusBoxes);
    assertEquals("Title should be the same","Fase", readStatusBoxes[0][0]);
    assertEquals("Table names should be the same","Ms_O2Wd,Il_Wd", readStatusBoxes[1][1]);
    assertEquals("Link name should be the same","process/returnsludge", readStatusBoxes[2][2]);

    readStatusBoxes=null;
```

```
        readStatusBoxes = xmlReader.readStatusBoxes(new FileInputStream(inputXML), "Data kvalitet - DQC");
        assertNotNull("Should not be null", readStatusBoxes);
        assertEquals("Title should be the same","NH4 LT1", readStatusBoxes[0][0]);
        assertEquals("Table names should be the same","DqAer1_No3", readStatusBoxes[1][1]);
        assertEquals("Link name should be the same","process/phase", readStatusBoxes[2][2]);

        List<String> mobileNumberOfBox = xmlReader.getMobileNumberOfBox(new FileInputStream(inputXML));
        assertEquals("Number of boxes should be the same",4, mobileNumberOfBox.size());
        assertEquals("The title should be the same","Fase", mobileNumberOfBox.get(1));
    }

}
```

Bilag K. Koden på en CD