# JohannesFog Carport Rapport



**Deltagere**

* + **Said Kayed,** [**cph-sk249@cphbusiness.dk**](mailto:cph-sk249@cphbusiness.dk)**, saidkayed, A**
  + **Sercan Atici,** [**cph-sa268@cphbusiness.dk**](mailto:cph-sa268@cphbusiness.dk)**, sercanatici, A**
  + **Tobias Jan Jensen,** [**cph-tj227@cphbusiness.dk**](mailto:cph-tj227@cphbusiness.dk)**, TobiasJensen1, A**
* Projektet og rapporten *blev nedskrevet 07-05-2018*

### Links:

* <https://github.com/saidkayed/Fog>
* <http://138.68.64.114/Fog-0.1-Test/>

## Indledning

Virksomheden Johannes Fog vil have deres it-system “byg-selv-carport” opgraderet. Vi har udviklet et system hvor kunden selv kan vælge mål og få vist en plantegning. Medarbejderen kan se ordrerne for alle kunder og har mulighed for at acceptere og se ordrernes styklister.

### 

Contents

[JohannesFog Carport Rapport 1](#_Toc515546144)

[Links: 1](#_Toc515546145)

[Indledning 1](#_Toc515546146)

[Baggrund 3](#_Toc515546147)

[Teknologivalg 3](#_Toc515546148)

[Krav 3](#_Toc515546149)

[Overordnet beskrivelse af virksomheden 4](#_Toc515546150)

[Arbejdsgange der skal IT-støttes 5](#_Toc515546151)

[Scrum user stories 7](#_Toc515546156)

[Domæne model og ER diagram 9](#_Toc515546157)

[Navigationsdiagram 11](#_Toc515546158)

[Særlige forhold 14](#_Toc515546159)

[Udvalgte kodeeksempler 15](#_Toc515546160)

[Status på implementation 20](#_Toc515546161)

[Test 21](#_Toc515546162)

[Process 22](#_Toc515546163)

[Arbejdsprocessen faktuelt 22](#_Toc515546164)

[Arbejdsprocessen reflekteret 23](#_Toc515546166)

### 

### 

### Baggrund

* Johannes Fog er en virksomhed hvor du kan forvente kvalificeret rådgivning til den bedste løsning. Fog hjælper deres kunder med at få overblik og komme i mål med deres byggeprojekt. Lige fra hvad der er godt at vide til vejledning i valg af materialer.
* Kunden skal kunne bestille en carport efter valgte mål.
* Kunden skal kunne se en plantegning af carporten.
* Kunden skal kunne se en stykliste af carporten.

### Teknologivalg

* Netbeans IDE 8.2
* MySQL server v14.14
* Ubunto 16.04.3 - linux server
* Tomcat 8

## Krav

Der er to dele i dette afsnit:

* Virksomhedens mål er at få solgt så mange Carporte som mulig og gøre det så nemt at bestille produktet for kunden, uden nogen form for forhindringer.
* Scrum user stories (product backlog)

Medarbejder delen:

* Som medarbejder kan jeg login.
* Som medarbejder kan jeg se ordrerne.
* Som medarbejder kan jeg godkende ordre.
* Som medarbejder skal jeg kunne slette en ordre.
* Som medarbejder vil jeg kunne fjerne materialer.
* Som medarbejder vil jeg kunne tilføje materialer.
* Som en medarbejder vil jeg gerne kunne se en stykliste af en carport.
* Som en medarbejder skal jeg kunne udregne pris til kunden på baggrund af styklisten.
* Som medarbejder vil jeg gerne kunne søge på kundes bestillinger.
* Som en medarbejder vil jeg gerne kunne se en genereret stykliste af en. carport for at kunne give kunden et tilbud.
* Som medarbejder vil jeg kunne se den samlede pris på carporten.
* Som medarbejder vil gerne have at styklisten er komplet.

Kunde delen:

* Som kunde vil jeg kunne registrere mig som kunde
* Som kunde vil kunne Indskrive mål på et ønsket carport.
* Som kunde vil jeg gerne kunne se status på min(e) ordre(r).
* Som kunde vil jeg kunne se styklisten på carporten efter den er godkendt.
* Som kunde vil jeg gerne kunne indtaste målene på en carport og herefter få vist en tegning.
* Som kunde vil jeg gerne kunne demo systemet uden login.
* Som kunde og medarbejder vil jeg gerne have at systemet er brugervenlig og driftsikkert.

### 

### Overordnet beskrivelse af virksomheden

Johannes Fog tilbyder køb af træ, byggematerialer og alt det du behøver til hus og have inden for f.eks. maling, bad og VVS, beslag, elartikler og lamper samt haveredskaber, grill og havemøbler. Hos Johannes Fog får du kvalitet og den rådgivning, du har brug for.

### 

### 

### 

### 

### 

### 

### 

### Arbejdsgange der skal IT-støttes

### 

### 

### Vores use case diagram giver et overblik over hvad man som kunde, ikke registreret kunde og employee kan i systemet.

### https://lh5.googleusercontent.com/smkF0fBwH4sp5WVdoGK9A-NXf2N1porH_TnuGAGzUSmVgZQntlbQ_3OhUMG8wCRXLBHkyMbUdAkGncmd46HCa5LFkUr3_XKlcvM85C5dXHoZ3js53fnoO3epQMm1zSY2uxYXkicu

### Aktivitets diagrammet beskriver arbejdsgangen for kunde og medarbejder, for at lave en ordre. Kunde logger ind, indtaster sine ønskede mål og insender den ordre. Herefter gennemgår medarbejderen styklisten, samt snakker med kunden. Herefter kan kunde acceptere eller afslå tilbuddet.

### 

### 

### 

### Scrum user stories

|  |  |  |  |
| --- | --- | --- | --- |
| How to demo | User Stories | Estimat(dage) | Tasks |
| 1 Kan ses under tabel | Som medarbejder kan jeg se ordrerne. | 1 | 1. Oprettelse af database. 2. Oprette userMapper metoder, klasser og jsp sider. |
| 2 Kan ses under tabel | Som kunde vil jeg gerne kunne indtaste målene på en carport, og herefter få vist en tegning. | 3 | 1. Jsp sider med inputs 2. Jsp side som tegner carporten vha svg |
| 3 Kan ses under tabel | Som en medarbejder, vil jeg gerne kunne se, en stykliste af en carport. | 6 | 1. Regnemotor 2. Jsp side som viser styklisten |
| 4 Kan ses under tabel | Som kunde vil jeg gerne kunne demo systemet uden login. | 1 | 1. Oprettelse af bruger “SystemTester” 2. Ændring af knap i viewsvg.jsp hvis brugeren er SystemTester |
|  | Som kunde og medarbejder vil jeg gerne have at systemet er brugervenlig og driftsikkert. | 6 | 1. Oprette fri navigering fra alle relevante sider til alle relevante sider. 2. Sikre alle inputs 3. Style hjemmeside |

1. "url" Bruger er logget ind og Bruger trykker på knappen "View Orders". Bruger ser en liste over ordrerne.

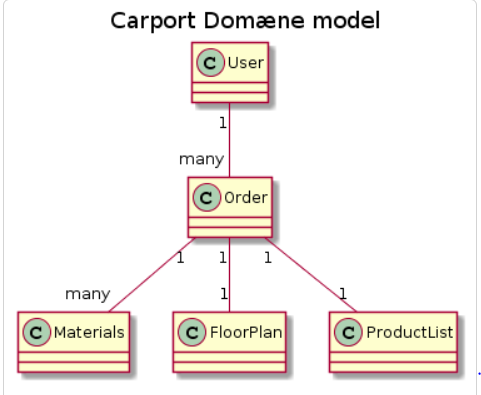
2. Kunden logger ind ved at skrive email og password. Kunden trykker på knappen "View Orders". Kunden indtaster mål på carport og trykker på knappen "View Plan". Kunden får en tegning vist.

3. Medarbejderen logger ind ved at skrive email og password. Medarbejder trykker på knappen "View Orders". Medarbejderen kan se en liste over alle ordre, medarbejderen trykker på knappen “Show Productlist”, Medarbejderen kan så se stykliste.

4. Kunden trykker på “Test System” Kunden indtaster mål på carport og trykker på knappen "View Plan". Kunden får en plantegning af carporten vist. Kunden skal herefter registrere og logge ind for at benytte systemet videre.

## 

## Domæne model og ER diagram



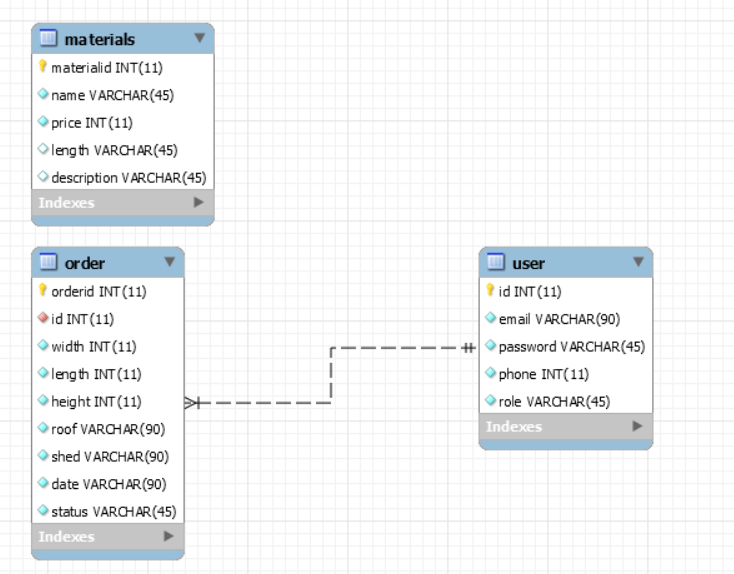
Relationer:

En User til mange Order, da en User kan have flere Orders uafhængigt af hinanden.

En Order til mange Materials, da en Order holder et array af mange Materials.

En Order til en FloorPlan, da hver Order kommer med en plantegning der er frit tilgængeligt.

En Order til en ProductList, da hver Order kommer med en productlist, som er tilgængelig efter at Orderen er blevet accepteret af en medarbejder.

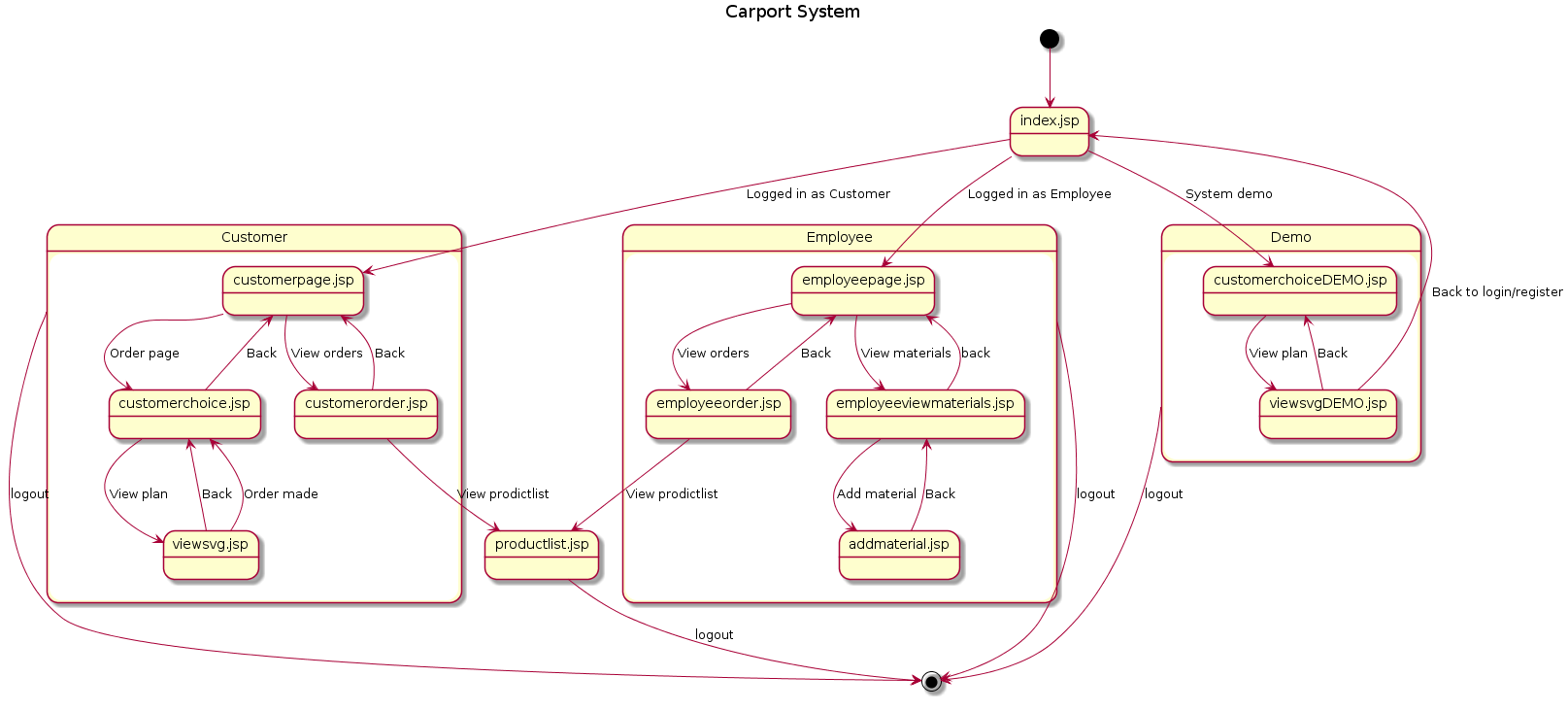


Vi har oprettet tre tabeller inde på databasen, hvor User har en til mange relation med Order. En User kan have en eller flere Ordrer.

I User tabellen har vi valgt at vores primary key “id” skal autogenereres, så hver eneste bruger der bliver oprettet inde på databasen er unikt. Det er vigtigt da vores Order tabel skal kunne vide, hvilke Ordrer som hver eneste kunde, har bestilt. Ordrer tabellen kendes ved at den har en foreign key “id” fra Useren.

Materials tabellen skal kunne holde på vores materialer, som bliver brugt til at bygge en carport. Tabellen er vigtigt, da vi bruger informationerne fra tabellen, inde på netbeans, til at udvikle styklisten.

## Navigationsdiagram



Vi har en fælles navigations bar i toppen af alle sider, når en bruger er logget ind. Den viser brugerens mail og holder en logout knap, som fjerner brugeren fra sessionen og sender til index, hvor en ny bruger kan logge ind/registrere eller demo systemet.

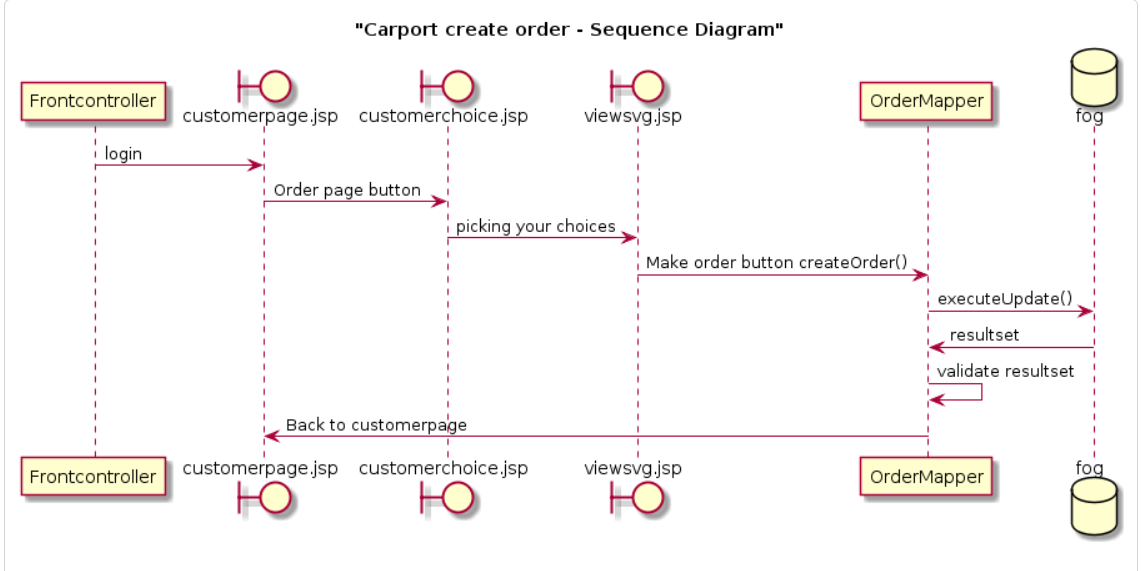
Customer feltet kan kun nåes af en customer, employee feltet af en employee og Demo feltet af en systemtester.

Jsp Sider:

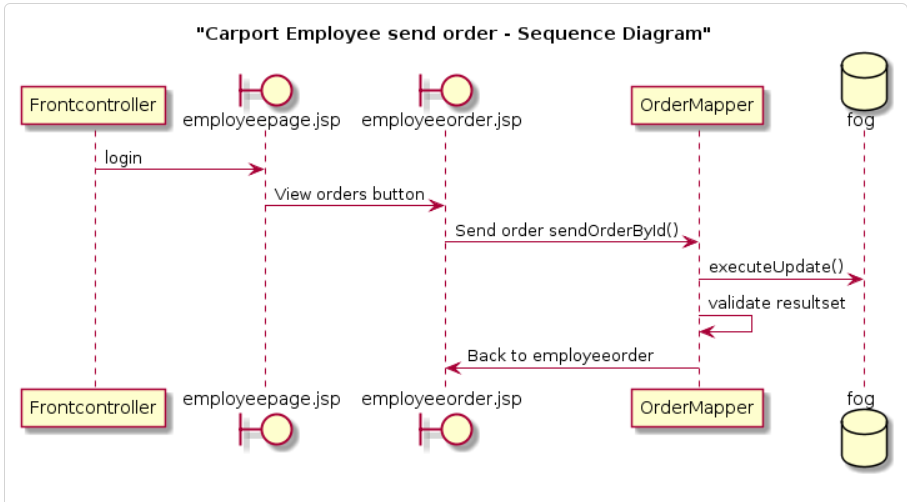
index, customerpage, customerchoice, customerorder, employeepage, employeeorder, employeeviewmaterials, addmaterial, viewsvg, productlist.

Navigation mellem jsp siderne håndteres i vores FrontController.

Sekvensdiagrammer



Carport create order sekvensdiagrammet giver os et indblik i hvordan en ordre bliver oprettet i it-systemet. I starten bliver vi logget ind og sendes til customerpage.jsp hvor vi kan klikke på knappen “Order page”. Dette sender dig videre til customerchoice.jsp. På denne side kan kunden vælge mål og klikke videre for at se en plantegning. Hvis kunden så ønsker produktet kan han/hun bestille ordren. Når kunden laver ordren bliver den kørt ved hjælp af vores createOrder() metode, hvor ordren bliver tilføjet til databasen. Kunden sendes derefter tilbage customerpage.jsp.



Carport Employee send order sekvensdiagrammet giver os et indblik i hvordan en ordre stykliste bliver sendt til kunden i it-systemet. I starten bliver vi logget ind og sendes til employeepage.jsp hvor vi kan klikke på knappen “View Orders”. Dette sender os videre til employeeorder.jsp hvor man finder en tabel med ordre. Medarbejderen kan på denne side klikke på knappen “Send Order”. Ved hjælp af sendOrderById() metoden bliver status opdateret i databasen. Kunden vil nu få adgang til knappen “Show Productlist”, hvilket betyder at de nu kan se deres stykliste.

## 

## Særlige forhold

* Hvilke informationer gemmes i session

I vores system gemmes useren som Object i sessionen efter login, hvad enten det er en customer, employee eller systemtester. Dette bruger vi til at vise nyttig information for brugeren på vores jsp sider.

* Hvordan håndterer man exceptions.

Vi håndterer exceptions igennem vores Front Controller, hvis denne bliver kastet bliver brugeren sendt til index hvor han får en besked om hvilken exceptions der er blevet kastet. Samtidigt logger vi hver gang en exception opstår

* Hvordan man på har valgt at lave brugerinput validering

Vi har lavet brugervalidering vha. HTML, for at sikre os mod at alle input er af rigtig “type” (tal/bogstaver) og alle er udfyldt, for at undgå tomme eller forkerte input til database.

* Hvordan man har valgt at lave sikkerhed i forbindelse med login

Vi tjekker at brugerens mail er unik og at password matcher denne mail. Vi bruger Preparedstatements for at sikre mod sql injections.

* Hvilke brugertyper der er valgt i databasen, og hvordan de er brugt i jdbc

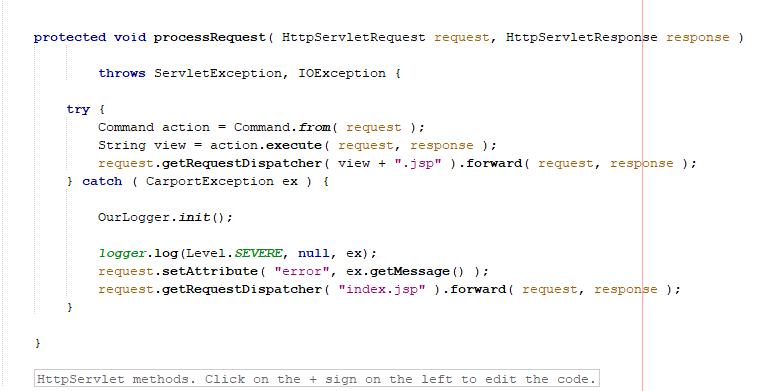
Vi har én bruger tob som kun har adgang til vores fog og fogdbtest sql skema. Denne kan oprette/registrere/slette og redigere frit inden for fog og fogdbtest skemaerne.

## 

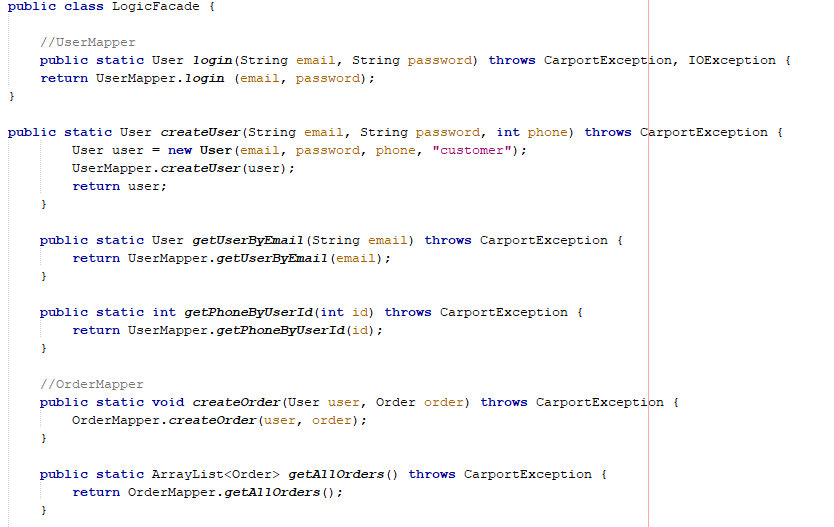
## Udvalgte kodeeksempler

Vi benytter tre design patterns, til at simplificere vores kode, Front Controller, facade og command patterns.

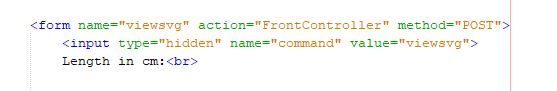
Front Controlleren sørger for vi får et meget nemmere navigation eller med andre ord et centraliseret kontrol. Da vi helst vil undgå at oprette en masse servlets, da det kan spare os en masse tid og give os et bedre overblik. Vi følte fra tidligere projekter at overblikket til tider glippede for os. Dette hjalp Frontcontrolleren os i høj grad med at bevare. Det er også igennem Frontcontrolleren at vi håndtere og logger vores exceptions.

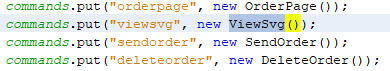


Facade pattern holder på alle vores mapper metoder og samler det i et sted. Det gøre det nemmere at bruge/redigere, mere fleksibilitet, og læseligt.



Command pattern, gør at den kan gemme metoder som value i et hashmap, de kan blive aktiveret hvis vi f.eks kalder på dem, ved at trykke på et knap på en jsp side. Hvis vi f.eks. tager udgangspunkt i, at vi bestiller en carport, så indtastes målene og der trykkes på viewplan knappen. Det sender de valgte mål ind til vores front controller, og fra front controlleren videre til command klassen vha. command from metoden. Her finder den keyen i hashmappet der matcher hidden input navnet fra jsp siden som vi sender videre med hidden input.





Disse klasser er abstrakte og ejer execute metoden som returnerer en string. Her kan vi hente og gemme informationerne der skal bruges og sidst returneres en string som er stien til den jsp vi vil til. Kan vores hidden input ikke identificeres kastes en exception og vi bliver sendt til index.jsp.



Vores regne motor er opdelt i flere dele for at give overblik og hjælpe mht. rettelse og videre implementationer. Vi følte at det havde været rart med et PO møde afsat til regne motoren alene hvor vores PO kunne sætte nogle regler, men da dette ikke var tilfældet har vi søgt på nettet efter hvad lignende hjemmesider bruger. Vi har så opbygget vores regne motor ud fra hvad vi har kunne finde hos andre.

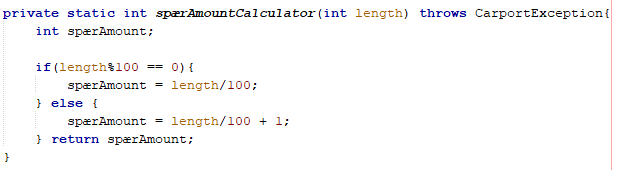
Der er blevet lavet mange ændringer undervejs og vi føler at der kan implementeres mange ting samt ændringer, men alle disse er i sidste ende nemme at udvikle på det skelet vi har.

Vi har sagt at vores planker er 20cm\*10cm\*x, hvilket nemt ville kunne ændres.

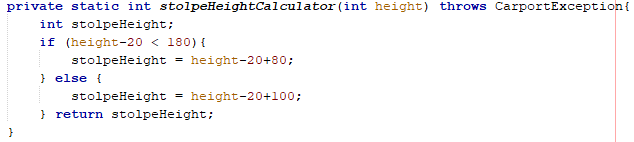
Vi har opdelt regne motoren i omkreds, spær, stolpe, tag og skur.

I vores omkreds bruges planker af 5 meter, da vores mål ikke overskride dette vil der altid blive brugt 4 planker til omkredsen. Har vi mål der overskrider dette kunne vi tilføje planker af 2 meter og lignende for at opfylde dette ønske. Til disse 4 planker bruges der altid 16 skruer, 4 til hver planke.

Vores spær calculator bruger en privat metode til at udregne hvor mange der skal bruges. Vi har sagt at der skal være et spær hver meter. Kan carporten ikke opdeles i præcist meter antal, tilføjes der en, så vi sørger for at der mindst er et spær pr meter. Vores spærplanker er også af 5 meter. Til spær bruger vi 2 skruer pr spær.



Stolpe calculatoren er opbygget meget magen til vores spær calculator. Her sætter vi blot en stolpe hver 2 meter i stedet. Yderligere har vi lavet en privat metode som udregner højden på stolperne ud fra regler vi fandt hos en anden carport udbyder angående hvor langt ned i jorden stolperne skal stå. Ideen her er at stolperne skal ned under frostfri dybde, hvilket er 80 cm. Dog skal stolper over 1.8m, 1 meter ned i jorden. Til stolper bruger vi 4 skruer pr stolpe.



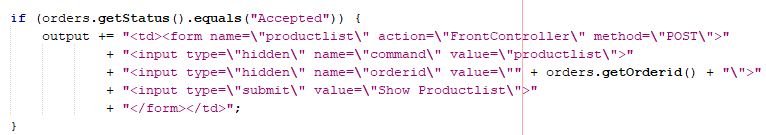
Vores tag calculator skelner mellem fladt tag og tag med rejsning. Har vi en fladt tag udregnes m2 blot, da de trapezplader vi bruger har en m2 pris. Til tag med rejsning har vi brugt en formel der altid har en rejsning på 10 grader, og herefter udregner vi igen hvor mange m2 trapezplader der skal bruges. Som support til taget med rejsning har vi planker af 5 og 2 meter. Vi bruger højde og sidestøtte ved hvert spær. Hvis højden er under 2 meter benytter vi 2 meters planker til højden og 5 meters planker til siderne, hvis ikke bruger vi 5 meter planker til det hele. Her bruges 4 skruer pr planke. Til trapezpladerne bruges 4 skruer pr trapezplade.

Vores skur calculator skelner mellem intet skur, 1x1 meter skur og 2x2 meter skur.

Her regner vi arealet af planker der skal bruges og trækker det område døren bruger fra. Vi bruger her planker af 2 meter, som skal skæres i 1 meters planker hvis skuret af 1x1 meter er valgt. Her bruges 4 skruer pr planke.

Priserne er ikke undersøgt nøje og er blot vejledende, da vi føler det er mere realistisk at det skulle aftales med en PO. Dette er dog simpelt at ændre.

For at kunden får den færdige stykliste er det aftalt med vores PO at en medarbejder skal godkende bestillingen. Fog’s ønske er at snakke med hver kunde for at specialisere hver bestilling og gennemgå hvad der er ønsket. Herefter kan betaling ske, og så har kunden adgang til styklisten. Dette gøres så kunden ikke blot kan se hvilke materialer der skal bruges. Styklisten tænkes dermed som en del af servicen, der betales for. Dette er også grunden til at denne feature er afskåret fra demoen. Kunden har derfor kun adgang til styklisten når betalingen er sket og ordren er accepteret.



## 

## Status på implementation

Vi føler at vi nåede alt der var prioriteret, vi havde selv 3 user stories som i sidste ende ikke blev prioriteret:

* Som kunde vil jeg gerne se en facadetegning.

Denne blev fravalgt af vores PO og blev derfor ikke implementeret.

* Som medarbejder vil jeg gerne kunne redigere i materialerne.

Denne blev først lavet sammen med vores user stories omkring at tilføje og fjerne materialer, men blev efter en dialog med vores PO fravalgt da det ville tage for lang tid at udvikle den i forhold til hvilken værdi den gav systemet og medarbejderen.

* Som medarbejder vil jeg gerne kunne ændre styklisten manuelt før den bliver sendt til kunde.

Denne kom sent på banen af vores PO og blev ikke prioriteret, vi valgte derfor at fokusere på driftsikkerhed og brugervenlighed frem for at implementere denne, selvom dens værdi er tydelig for både kunde og medarbejder.

Et af vores store problemer har opstået under deployment af projektet. Vi har det dog kørende som ønsket lokalt.

Vi fik også en anden fejl da vi ville lave j-unit tests for vores orderMapper, vi har derfor udeladt disse fra vores aflevering.

Vores regne motor kan også videreudvikles en del, men vores PO var tilfreds med den og valgte derfor at prioritere andre dele.

## 

## 

## Test

Vi har lavet et sql skema magen til vores fog skema der benyttes i vores system frem for en mockup connection. Denne opsættes på ny hver gang testen køres. Dette har vi gjort for at undgå at slette og redigere i vores primære skema hver gang vi ville køre vores test.

Vores hovedsagelige formål med vores tests var at sikre at vi havde connection til database

Mindre tests er også lavet løbende i vores main metode.

* Hvilke klasser er testet

userMapper

carportCalculator

* Hvilke metoder der er testet

userMapper.createUser

userMapper.login

carportCalculator.calculator

* Dækningsgrad af dine tests for de valgte metoder og klasser

Vi starter med at sikre os at vores connection kører. Herefter tester vi at vores userMapper.login virker som ønsket, både med rigtigt og forkert input. Til sidst tester vi at userMapper.login giver os en user med den ønskede rolle.

UserMapper.createUser bliver testet ved at vi laver en ny user, laver den med createUser og herefter henter den ud vha userMapper.login og sikrer os at disse to usere er identiske.

CarportCalculator.calculator kræver orderid, så her laver vi en ny order og kører så calculatoren på denne order hvorefter vi sikrer os at vi får en stykliste af korrekt størrelse og indhold, efter vores forventninger.

Dækningsgraden er ikke så stor, men vores hovedsagelige formål med vores tests var at sikre at vi har en connection til en database.

## 

## Process

### Arbejdsprocessen faktuelt

### Dette projekt var vores første møde med scrum. Som base for vores projekt startede vi med at lave både login og registrering. Den user story der blev prioriteret til første sprint var:

* Som medarbejder kan jeg se ordrerne.

Denne krævede selvfølgelig visse andre funktionaliteter for at kunne vise en ordre:

* Som kunde vil kunne Indskrive mål på et ønsket carport.

Den user story der blev prioriteret til vores næste sprint var:

* Som kunde vil jeg gerne kunne indtaste målene på en carport, og herefter få vist en tegning.

Dette krævede en dynamisk svg, som kunne lave en tegning med realistiske forhold, hvilket gav kunden et indtryk om forholdene. Denne tegning viste også skur hvis valgt.

Til 3. sprint valgte vores PO:

* Som en medarbejder, vil jeg gerne kunne se, en stykliste af en carport.

Dette krævede første udkast af vores regne motor, som på baggrund af en ordre skulle kunne generere et array af materialer. Dette krævede også udvikling af andre user stories:

* Som medarbejder vil jeg kunne fjerne materialer.
* Som medarbejder vil jeg kunne tilføje materialer.
* Som medarbejder vil jeg kunne se den samlede pris på carporten.

Når vores første udkast virkede, færdigudviklede vi så regne motoren:

* Som medarbejder vil gerne have at styklisten er komplet.

Vores 4. sprint gik ud på at lave en funktion med en demo:

* Som kunde vil jeg gerne kunne demo systemet uden login.

Sidste sprint gik ud på at sikre os at alt virkede som ønsket, samt styling af hjemmesiden:

* Som kunde og medarbejder vil jeg gerne have at systemet er brugervenlig og driftsikkert.

Scrum master rollen blev fordelt forholdsvist ligeligt.Vi prøvede alle at have rollen, hovedsageligt fordi det var vores første opgave med scrum. Scrum master rollen gik for os ud om at uddele ansvar og gøre det klart for alle, hvad der skulle udvikles.

Under første sprint var Tobias scrum master, og han stod derfor for fordelingen af de første arbejdsopgaver, senere var både Said og Sercan også scrum master under et sprint. Undervejs blev det også tydeligere hvilke værdier rollen gav os.

Vores 4. sprint gik ud på at lave en funktion på en demo. Dette kom fra en længere samtale med vores PO hvor vi til sidst blev enige om at login/registrering har det med at skræmme potentielle kunder væk. Dette kunne lindres ved at udvikle en demo funktion som gav kunder muligheden for at fornemme nogle af funktionaliteten uden at skulle registrere sig. Dette var første gang vi fornemmede hvordan en samtale med PO kunne udvikle sig til en løsning. Hvor vi i de tidligere sprints var blevet beordret til at udvikle, fandt vi i stedet her en fælles værdi i en funktionalitet som vores PO var enige med os i. Det var først her vi så værdien i at føre en samtale, og selv komme med et bud om hvad hjemmesiden kunne bruge, frem for at følge ordre, som vi havde gjort indtil da.

I alle hverdage hvor vi arbejde på projektet startede vi dagen med et hurtigt møde. Formålet med dette var at få alle på lige fod mht. hvad der skulle arbejdes på, samt hvor langt de forskellige dele var udviklet. Vi har dog gennem hele projektet arbejdet ved siden af hinanden, så der var aldrig stor tvivl om hvordan projektet skred frem, men vi så alligevel værdien i at starte dagen ud med et statusmøde.

Vi afholdte retrospectives efter hvert sprint. Formålet her var at være selvkritiske og få rettet nogle af de ting vi gjorde forkert. Det er klart at vi med tiden blev bedre til scrum, og det hjalp os her at snakke om nogle af de fejl og mangler vi havde i sprintet før, for at kunne udlede dette til de fremtidige sprints. Vi afholdte disse møder mellem alle sprints.

### Arbejdsprocessen reflekteret

Scrum master rollen fungerede ikke optimalt for os, vi kunne godt se ideen med rollen, men i et 3 mands team hvor alle var lige meget med, var det nemmere at “deles” om rollen og aftale tingene imellem os. For øvelsens skyld tog vi dog scrum master hatten på til mindst et sprint hver, men dens rolle var ikke så tydelig i sidste ende.

Det vi hovedsageligt fokuserede på under vores retrospektive møder var dels hvad vi gjorde forkert, men også hvad der fungerede rigtig godt. Denne dialog gjorde det også klart for os alle om hvad vi skulle klø på med, og hvad vi skulle vende os af med. Eksempelvis gik der for lang tid før vi startede på branches. Dette betød at vi tidligt i forløbet kun arbejde på en computer. Et af vores møder efter at vi startede på branches lagde derfor stor vægt på hvor dumt det var at vi ikke havde benyttet os af dette tidligere.

Vi følte at nedbrydningen af user stories til tasks gik ret godt. Vi reflekterede over opgaven og delte det derefter ret hurtigt user storien i små dele, som vi fordelte i blandt os. Dette gik naturligvis mere efter hensigten da branches blev brugt.

Vi følte at der til tider var lidt uoverensstemmelse mellem vejledning og PO møder. Vi blev blandt andet bedt om til vejledning at have både plantegning og facadetegning, hvor vores PO kun lagde vægt på plantegningen. Her valgte vi at følge vores PO, og skrev derefter de user stories som vores vejleder, samt os selv så værdi i, men ikke blev valgt af PO ind i vores meistertask, dog uden at udvikle dem.

Det tog et par sprints for os at finde en rytme der var produktiv. Det var først efter 4 sprint vi virkelig havde fået fat på hvordan dialogen mellem udvikler og PO kunne give bedre resultater end blot at følge ordre. Selvom det i starten var uklart hvad scrum gik ud på, kunne vi undervejs og specielt til sidst se værdien i scrum.

