Fog Carporte

Maj 2020



|  |  |
| --- | --- |
| **Datamatiker 2. Semester, Hold E, Gruppe 666** | |
| **Alexander Pihl**  *Email:* [cph-as509@cphbusiness.dk](mailto:cph-as509@cphbusiness.dk)  *Github:* <https://github.com/AlexanderPihl> | **Morten Rahbek**  *Email:* [*cph-mr462@cphbusiness.dk*](mailto:cph-mr462@cphbusiness.dk)  *Github:* [*https://github.com/amazingh0rse*](https://github.com/amazingh0rse) |
| **Mick Larsen**  *Email:* [*cph-ml616@cphbusiness.dk*](mailto:cph-ml616@cphbusiness.dk)  *Github:* [*https://github.com/MivleDK*](https://github.com/MivleDK) | **Per Kringelbach**  *Email:* [cph-pk171@cphbusiness.dk](mailto:cph-pk171@cphbusiness.dk)  *Github:* <https://github.com/cph-pk> |
| **Jean-Poul Leth-Møller**  *Email:* [*cph-jl360@cphbusiness.dk*](mailto:cph-jl360@cphbusiness.dk)  *Github:* [*https://github.com/Jean-Poul*](https://github.com/Jean-Poul) | |

Indhold

[0. Projektformaliteter 3](#_Toc41589564)

[1. Indledning 4](#_Toc41589565)

[2. Baggrund 4](#_Toc41589566)

[3. Teknologivalg 5](#_Toc41589567)

[Programmering og syntaks 6](#_Toc41589568)

[Frameworks og teknologier 6](#_Toc41589569)

[Software 6](#_Toc41589570)

[4. Krav 6](#_Toc41589571)

[4.2 Scrum user stories 7](#_Toc41589572)

[4.1 Overordnet virksomhedsbeskrivelse 8](#_Toc41589573)

[4.2 Arbejdsgange der skal IT-støttes 8](#_Toc41589574)

[5 SCRUM Userstories 9](#_Toc41589575)

[6 Diagrammer 13](#_Toc41589576)

[6.1 ER Diagram 13](#_Toc41589577)

[6.2 Navigationsdiagram 14](#_Toc41589578)

[7 Særlige forhold 15](#_Toc41589579)

[8 Udvalgte kodeeksempler 15](#_Toc41589580)

[9 Status på implementation 16](#_Toc41589581)

[10 Test (Husk word fil fra Jon her og indrag v-model) 17](#_Toc41589582)

[11 Process 18](#_Toc41589583)

[11.1 Arbejdsprocessen faktuelt 18](#_Toc41589584)

[11.2 Arbejdsprocessen reflekteret 18](#_Toc41589585)

[12 19](#_Toc41589586)

[13 19](#_Toc41589587)

[14 Appendix 19](#_Toc41589588)

[14.1 User Stories 19](#_Toc41589589)

[Sprint 1 19](#_Toc41589590)

# 0. Projektformaliteter

Projektets kildekode: <https://github.com/Jean-Poul/Eksamensprojekt_Fog>

Projektets Java Doc: a

Produktets demovideo: a

Produktets website: <http://134.209.243.195:8080/Fog/>

Login: admin@fog.dk

Kodeord:

# 1. Indledning

Vi har fået stillede til opgave at kunne hjælpe virksomheden Johannes Fog med at udarbejde et IT-system, hvor man kan bestille en standard og byg selv carport. Vi har valgt at gøre det via et website, som har forbindelse til en dedikeret server med en database, så fog kan hente, gemme og ændre i deres data, samt danne sig et overblik over deres data. Fog skal kunne tage imod forespørgsler fra kunder, som en medarbejder, herefter kan se og rette i. Herfra kan der udarbejdes et acceptkriterie, som herefter kan ses af kunden. Vi har i vores gruppe haft fokus på, at som en kunde skal man selv kunne vælge mål, materielle, tagvinkel og skur med tilhørende mål. Vi har dog forholdt os til at højden er fastlagt på 2.10 meter. Alt bliver udregnet ud fra Fogs priser og der bliver også udregnet alt materielle, som skal bruges for at kunne konstruere sin carport. Mere om dette i afsnittet *arbejdsgange der skal IT-støttes*.

Fogs website skal være dynamisk og vil blive kodet ved hjælp af bl.a. HTML, CSS, Bootstrap, Java og SQL for til sidst at kunne afvikle det igennem en tomcat webcontainer. Mere om dette i afsnittet *teknologi valg*. Når der siges dynamisk, menes der, at man kan oprette, læse, opdatere og slette data fra en database (CRUD). Databasen normaliseres på 1, 2 og 3. normalform og vi forholder os ikke til 4 og 5 normalform i dette forløb.

Vi er blevet rådgivet til at skulle udvikle det endelige produkt ud fra en skabelon, som vi har fået tildelt. Denne skabelon bruger et design mønster kaldet MVC, som er et framework, der står for Model, View og Controller. Det bruges i programmering til at adskille ens kode, for at gøre det mere overskueligt og nemmere at vedligeholde. Det endelige produkt skal uploades og køre på en droplet hos udbyderen Digital Ocean. Vores kildekode er tilgængelig på GitHub, hvor vi med hjælp af javadocs har gjort det nemmere, at danne sig et overblik over projektets indhold.

Opgaven er opdelt i to dele med en administrator del og en kunde/bruger del. Johannes Fog ønsker en administrator side, hvor man skal kunne se en oversigt over alle forespørgsler og hvor en forespørgsel kan rettes og godkendes. Herudover kan en administrator rette i standardmål, samt priserne til materiellet.

# 2. Baggrund

Virksomheden Johannes Fog har fokus på selv byg og har derfor flere byggecentre, hvor kunder kan komme for at få rådgivning til deres egne bygge projekter. Fog står for kvalitets vare og udover deres byggecentre har de et boglig og designhus, hvor design vare er til udstilling for at kunne få inspiration og vejledning. Fog brander sig med at de har et stort udvalg med fokus på kvalitet og vælger kun produkter, som dækker kundens brugsbehov.

Kunder som ville gøre brug af websitet, ville kunne stille følgende krav:

* Som kunde kan jeg forespørge en carport med standardmål
* Som kunde kan jeg forespørge en carport med udvalgte mål
* Som kunde kan jeg tilføje et skur til min carport med udvalgte mål
* Som kunde kan jeg tilføje grader på mit tag
* Som kunde kan jeg selv vælge mit materiale
* Som kunde forventes der at få tilsendt status på ens forespørgsel
* Som kunde forventes der at Fog tilsender en oversigt af materiale
* Som kunde forventes der at Fog tilsender arbejdstegninger

# 3. Teknologivalg

Vores Projekt er programmeret i Java under et Maven framework. Alle i teamet har gjort brug af Windows OS.

Vores program tager udgangspunkt i model-view-controller, som er et software design mønster, der er til for at opdele ens kode i tre elementer. Model er til for at gemme data, som skal fremvises ellers behandles. View er til for at fremvise et output, hvilket gøres på vores jsp sider. Controller opfanger et HTTP request og sender et HTTP response tilbage.

Vi har fået tildelt en skabelon, som vi skulle forholde os til og bruge som rygraden af vores projekt. Her er de mest essentielle klasser Command, FrontController og brugen af Servlets. Command opretter en instans af et HashMap. Kendetegnet ved et HashMap er at det indeholder en nøgle og en værdi. Det er igennem denne funktion vi gemmer en reference til vores java klasser og kan derfor lave en instans af de ”rigtige” klasser. FrontController klassen er til for at lave en instans af Command og til at sende programmet videre til en jsp side, som er givet i form af et request fra en bruger. Sidst men ikke mindst gør vi brug af HTTPServletRequest og HTTPServletResponse, som nedarver fra Servlet klassen. Dette gøres fordi vi arbejder med et request fra en bruger, hvorefter der tilsvarende sendes et response tilbage. Alt dette kan bruges da vi bruger et webapplikationsframework, som i vores tilfælde hedder tomcat.

Hele projektet er versionsstyret med git bash som bindeled til Github. ”Milepæle” er fusioneret til en Master-branch imens løbende udvikling er styret på en Production-branch.

Der er desuden udarbejdet en omfattende dokumentation i javadocs, som kan ses på projektets github side.

Følgende teknologier er blevet anvendt for at kunne udarbejde projektet og holde kommunikationen intakt i vores gruppe.

|  |  |  |
| --- | --- | --- |
| Programmering og syntaks  * CSS3 * JSP * HTML5 * Java - 1.8.0\_251 * Javascript (ECMAScript 2015) * MySQL - 8.0.18 * PlantUML | Frameworks og teknologier  * Apache Tomcat - 9.0.31 * Bootstrap - 4.3.1 * JavaE Web API - 7.0 * JSTL - 1.2 * JUnit - 4.12 * Maven compiler plugin - 3.1 * Maven dependency plugin - 2.6 * Maven war plugin - 2.3 * MySQL Connector Java - 8.0.19 | Software  * Adobe XD - 29.0.32 * Discord - 0.0.306 * Draw.io - 11.3.0 * FileZilla Client - 3.48.1 * Firefox - 76.0.1 * Git bash - 3.0.2 * Google Chrome - 81.0.4044.138 * IntelliJ IDEA (IDE) - 2020.1.1 * MySQL Workbench - 8.0.18 * Photoshop - 20.0.7 * PuTTY - 0.73 * Ubuntu OS 19.10 * Windows 10 OS Build 18362.836 * winRAR - 5.80.0 * Word for Microsoft 365 * Zoom - 5.0.2 |

# 4. Krav

De følgende sektioner har til formål at give et indblik i hvad virksomheden Johannes Fog forventer og ønsker af deres nye IT-System, samt give et indblik i hvad deres nuværende system har af mangler.

Der vil også blive henvist til hvordan Fog, som virksomhed, vil få gavn af vores system og få optimeret deres arbejdsgange.

***Herudover vil der blive vist hvilke user stories, vores gruppe har arbejdet med for at kunne nå i mål med vores produkt. User stories er oprettet ved hjælp af møder med en product owner og vedligeholdt ved hjælp af den agile udviklingsmetode kaldet scrum.***

# 4.1 Overordnet virksomhedsbeskrivelse

Vi vil ligge ud med at komme ind på, hvilke håb virksomheden Johannes Fog, har med vores produkt. Fog har et ønske om at kunne få opdateret deres IT-system, da deres nuværende system er udviklet i 1999, hvilket må siges at være gammelt på nuværende tidspunkt, og specielt indenfor IT-branchen, da alt bevæger sig meget hurtigt i denne verden. I det nuværende system er der endda ikke mulighed for at opdatere data, da personen som udviklede systemet for Fog, ikke længere arbejder hos dem og udviklede det ikke med henblik på, at det skulle være relevant på daværende tidspunkt. Den tidligere ansat, som udviklede systemet, havde en virksomhed ved navnet Bergman IT-service, som må antages ikke at eksistere længer, da Fog ikke har haft mulighed for at opdatere deres software. Fog har dog forsøgt sig med at få opdateret deres nuværende system, hvilket kostede dem i omegnen af 1 million kr., men da det ikke kunne tale med det gamle system, var de nødsaget til at bibeholde deres nuværende system, hvilket må siges at have været en dyr læringsproces.

Fog kan forvente at få alle deres ønsker opfyldt ved brug af vores system, da vi har vedligeholdt de funktioner de er glade for, optimeret dem og endda tilføjet nye funktioner. Inden der vil blive forklaret hvordan vores system opfylder Fogs forventninger, vil vi først give et overblik over hvordan deres system virker på nuværende tidspunkt.

Fog ikke mulighed for at kunne rette data i deres nuværende system. Det vil sige at man som Fog medarbejder, kun har mulighed for at kunne ændre i den samlede pris ved hjælp af en dækningsgrad. Der er ikke mulighed for at kunne ændre i materialevalg, mål eller antal. Oven i det er der fejl i den tekst, som fortæller en medarbejder, hvilke mål og materiale, som skal bruges til en kundes forespørgsel på en carport. Det skal dog siges at selvom der er fejl i teksten virker systemet stadigt, som forventet, da de ansatte kender til denne fejl.

Når en forespørgsel på en carport bliver sendt til Fog, som bliver modtaget via e-mail, skal en medarbejder selv ind og genindtaste forespørgslens mål for at kunne generere en stykliste, arbejdstegning og pris til kunden. Herfra kan Fog tage stilling til om det er realistiske mål kunden forespørger og om det er noget kunden selv, har mulighed for at bygge. Da visse kunder ikke selv har mulighed for at bygge deres egen carport, har Fog kontakt til flere tømrer, som har været med til at bygge disse carporte i mange år, hvilket gør det nemt for Fog at udregne en kostpris. Det opnås kun pga. deres store erfaring indenfor faget.

Ved at gøre brug af vores IT-system ville Fog få en opdatering af deres system, samt få indfriet deres forventninger i form af en optimering af deres nuværende proces og håndtering af salg af deres carporte.

Vores system er udviklet på sådan en måde, at en kunde ikke har mulighed for at vælge urealistiske mål, da vi har indsat kriterier for diverse valg af mål, en kunde kan foretage sig ved deres forespørgsel. Derudover skal man ikke genindtaste en kundes forespørgsel for at få udregnet en stykliste med pris, samlet pris og arbejdstegninger, da systemet tager sig af alle disse ting lige så snart en kunde sender sin forespørgsel.

Til sidst skal der kort nævnes, at en medarbejder har mulighed for at rette i alt det data der høre til en forespørgsel, da vores system har fuldt adgang til en database, hvor man kan ændre i alt data, men man skal dog have de rigtige rettigheder og adgang. Det vil sige at det kun er tilegnet en administrator af systemet.

Alt i alt kan man sige at Fog forventer at digitalisere de daglige arbejdsgange, samt optimere de ansattes brug af deres tid, hvilket vil være med til at optimere virksomhedens omsætning.

## 4.2 Arbejdsgange der skal IT-støttes

Som nævnt tidligere, har virksomheden Johannes Fog givet os en opgave, hvor et nyt IT-system skal udarbejdes for dem. Fog har flere ønsker og forventninger til deres nye system.

Inden vi dog kan starte på at udvikle deres nye IT-system, skal der designes og udarbejdes et website, da vi på denne måde får mulighed for at binde vores system sammen med en dedikeret server, hvilket vil strømline hele deres proces og gøre det hele dynamisk. Vi har i dette tilfælde valgt at holde os tæt op af deres nuværende design og samme farvevalg, da det er Fogs identitet

Før vores IT-system blev udarbejdet for Johannes Fog var en bruger af systemet nød til at fysisk være til stede på lokationen af det installeret software, da det hele blev kørt på en lokal klient. Derudover havde bruger af systemet ikke mulighed for at kunne rette i information vedrørende deres udvalg som en kunde havde forespurgt og man kunne tilmed ikke rette i det udvalg Fog selv havde til rådighed i form af materiale, længde og antal. Pris havde en medarbejde selv mulighed for at tilrette, men der nævnes ikke om man havde mulighed for at rette i materiale priser. Kun at man kan rette i total prisen ved at ændre på dækningsgraden. Dette gjorde det meget statiks og derved var der ikke mange muligheder for at rette en eventuel fejl i ordren eller i systemets indhold. Ved udregninger af materiale, som f.eks. antal af stolper og spær tog Fogs system udgangspunkt i forældet tabel. Det var stadigt sikkert, efter Fogs menings, da deres udregninger af mål og tryk var foretaget med ”livrem og seler”. Til sidst skal det nævnes at de ikke havde mulighed for at kunne sende en status på en forespørgsel. Det vil sige at systemet ikke automatisk fik tilsendt en mail til en kunde angående deres forespørgsel på en ordre, eventuelle rettelser og tilbud, som Fog ville kunne give til kunden. Alt dette blev udført via et opkald, når en ansat havde set på en forespørgsel.

Hos Fog forventer man at der kan ses en oversigt over hvilke forespørgsler de har fra deres kunder. Hertil skal man have mulighed for at kunne se en kundes information. Derudover skal man kunne se status på deres forespørgsler, da det giver en Fog medarbejder mulighed for at kunne tage stilling til en forespørgsel. Grunden til, at der skal være et statusfelt, er at en medarbejder kan se hvor langt forespørgslen er i forløbet og ved hvornår en sælger eller byggemester skal kontakte kunden, med henblik på mersalg eller konstruktions rettelser. Fog nævner selv denne proces kan virke lidt gammeldags og langsomlig, men det er noget de godt kan lide at forholde sig til, hvilket vi ikke ville ændre på. Det vil sige at selvom alt i vores system er automatiseret, skal en medarbejde stadig ind og læse en forespørgsel for derved at kunne acceptere eller rette for at kunne opfylde kundens behov. Herfra kan man sende en kvittering til en kunde i form af en e-mail. Fog kan herved sideløbende ringe til kunden for at fastholde deres ønskede kontakt med henblik på mersalg og ændringer.

Da alt data kommer fra en database, har Fog også mulighed for at rette i deres data. Det vil sige at man som Fog medarbejde har jeg mulighed for at ændre i materialevalg, længde, antal og pris. Dette er muligt, da vi forholder os til de fire grundlæggende funktioner i en database applikation. Kort sagt kaldes det CRUD og står for create, read, update og delete.

Før en Fog medarbejder modtager en kundes forespørgsel bliver valg af materiale, længder, antal og pris udregnet, så en medarbejder ikke selv skal sidde og regne sig frem til det eller genindtaste mål, som er tilfældet på nuværende tidspunkt. Alt er udregnet efter byggeregler og efter en opdateret tabel med mål og tryk, så man kan være sikker på, at carporten holder når den er færdigbygget.

Efter udviklingen af det vores system skal man ikke længere være fysisk til stede, hvor systemets software er installeret, for at kunne ændre i systemet. Alt foregår ved hjælp af en dedikeret server og behøver ikke software installeret på en lokal klient. Alt en bruger af systemet har brug for er en forbindelse til internettet. Herudover er der mulighed for at kunne ændre i mål, materialevalg, materiale pris(stykliste), total pris og status på en forespørgsel, samt lave ændringer til det udvalg, som Fog vil kunne udbyde for en kunde.

# 5. SCRUM Userstories

Scrum er, som nævnt i introen af dette kapitel, en agil udviklingsmetode, der har stor fokus på ledelsen af projektet. Strukturen i scrum består i af at først har man et product backlog meeting med sin product owner, sprint meeting med sit team, daily meeting med sit team, sprint review meeting med sit team for så at afslutte hele ens forløb med retrospective meeting. Mere om dette i afsnit *Process.*

Under vores forløb har vi haft fokus på følgende user stories, fra vores product backlog, som er blevet godkendt af en product owner, der var tilknyttet vores gruppe. *Se* afsnittet *Appendix* for at få et overblik over alle vores user stories og de tilhørende tasks.

**1 Sprint**

#21 Kunde: Forespørgsel på spec. carport (basic)

#22 Kunde: Forespørgsel på spec. carport (basic m. tag)

#23 Kunde: Forespørgsel på spec. carport (basic, tag + skur)

#34 Kunde: Forespørgsel på materialetyper

**2 Sprint**

#5 Fog: Se forespørgsler

#15 Fog: Generering af tegning

**3 Sprint**

#80 Dynamisk view a SVG

#82 Fog: Se forespørgsel

#8 Fog: Slette forespørgsler

#14 Validering af spec. løsninger

#109 Kunde: Kvittering ved forespørgsel

**4 Sprint**

#11 Fog: Administrering af varekatalog

#7 Fog: spec. carport (Redigering)

**5 Sprint**

#171 Slutspurt

#6 Fog: spec. carporte (Modtaget tilbud)

#148 Fog: CRUD

#59 Fog: beregning af pris

Af de user-stories der er aftalt med product-owneren, har vi valgt at fremhæve og beskrive fire af dem, der hver især dækker en vigtig funktion i programmet.

Der er valgt følgende userstories:

* **#21 Kunde: Forespørgsel på spec. carport (basic)**
* **#15 Fog: Generering af tegning**
* **#11 Fog: Administrering af varekatalog**
* **#7 Fog: spec. carport (Redigering)**

#21 Dækker over at kunden skal kunne lave en forespørgsel på en carport i specialmål.

#15 Dækker over at der skal genereres dynamiske tegninger til en forespørgsel.

#11 Dækker over at Fog skal kunne ændre og opdatere deres varekatalog.

#7 Dækker over at Fog skal kunne redigerere en ekisterende forespørgsel, således at fejl og mangler kan ændres.

Resten af vores userstories kan ses i backloggen i appendix #.

**#21 Kunde: Forespørgsel på spec. carport (basic)**

*Som kunde*

Vil jeg kunne forespørge på en carport i specialmål (L,H,B) med standard tag, u. skur

Således at jeg kan modtage et tilbud på samme.

Accept kriterier:

Kunden skal kunne angive udvalgte mål til længde, højde, bredde samt vælge flat tag og fravælge skur.

Estimat:

Back end: L

How-to:

Home -> Bestil -> Byg Selv -> Indtast dine mål

Fra forsiden klikkes der på ”Bestil” i navigationsbaren i toppen så man lander på ”vælg carport” siden.

Derefter trykkes der på ”Byg selv” i højre kolonne nederst. Hvorefter man lander på byg selv siden.

Her kan man angive alle målene til sin carport, tag og skur samt sende forespørgslen afsted til fog.

**Tasks**

#30 Krav til dimensionering skal implementeres (Antal søjler, dimensioner mm)

#54 Kundeforspørgsels <select> skal populeres med korrekt data

*Vælg tag*

*1 - valg tagtype: fladt / rejst*

*2 - hvis fladt = kun ét slags tag*

*3 - hvis rejst = vælg hældning*

*4 - hvis rejst = vælg beklædning*

#39 Design konverteres til kode

*Mockup skal omdannes til .jsp*

#27 Database: E/R Diagram

*Der skal laves et E/R diagram over vores database*

#51 Beregning af lægter

*CarporpCalculation skal kunne beregne antallet af lægter og afstand*

#48 Database: Queries til tabelopslag på raft length/spacing/dimension

*Database queries til tabelopslag skal laves i datamapper*

#24 Opsætning af command-pattern skabelon og klargøring på git med MASTER og PRODUCTION branch

#26 Design: Hjemmeside design til index og customerpage

*Index skal laves og customerpage skal kunne håndtere bestillinger*

#37 Database og program skal forbindes (Med standard credentials)

*Connetor klassen & datamapper skal ordnes så der kan oprettes forbindelse til databasen*

#28 Domæne model (.uml)

*Første udkast af domæne modellen skal laves*

#29 Opstart på klasse diagram

Første udkast af klasse diagram skal laves

**#15 Fog: Generering af tegning**

*Som Fog-medarbejder*

Vil jeg kunne generere en tegning på kundens forespørgsel  
således, at jeg kan visualisere løsningen for kunden

Accept kriterier:

Dynamiske tegninger af kundens forespørgsler skal kunne generes og vises

Estimat:

Front end: L

Back end: XL

How-to:

Home -> Log Ind -> Log ind -> Åben -> Se tegning og stykliste

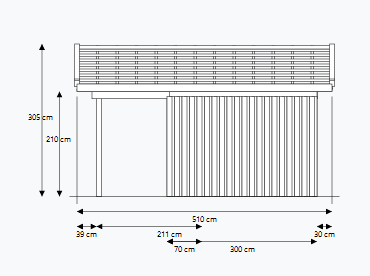
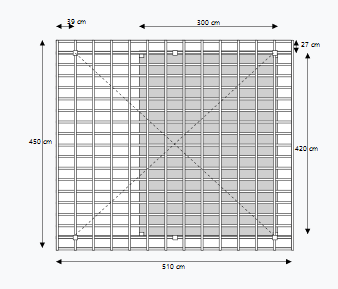
Fra forsiden trykkes der på Log ind i øverste højre hjørne. Derefter indtastes admin bruger info

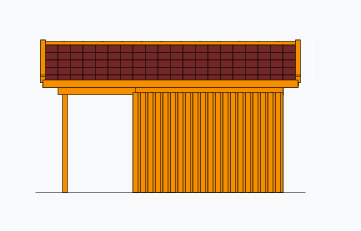
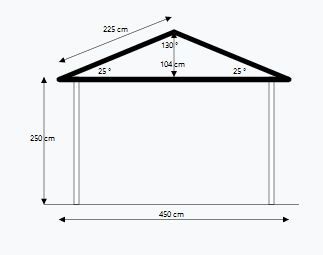
Brugernavn: admin

Kodeord: 1234

og der trykkes på log ind.

Herefter trykkes der på ”åben” ud for en forespørgsel i højre side. Nederst på forespørgsel siden trykkes der på ”se tegning og stykliste”. Hvorefter de dynamiske tegninger ses nederst på siden

Tasks

#66 SVG tegning forfra

*Der skal laves en dynamisk SVG tegning af carporten set forfra med mål og vinkler*

#64 SVG oppe fra

*Der skal laves en dynamisk SVG tegning set oppe fra med mål*

#63 SVG mål

*SVG klasserne skal kunne modtage mål fra carportcalculation*

#67 SVG tegning fra siden

*Der skal laves en dynamisk SVG tegning set fra siden med tag med mål*

#69 xd design til SVG landing page

*Der skal laves et design til den side der viser alle SVG tegningerne*

#56 Hente data fra database (dimensioner tabel) til program

*Der skal laves database queries til at hente forespørgsels data*

**#11 Fog: Administrering af varekatalog**

*Som Fog-medarbejder*  
Vil jeg kunne opdatere varer og priser i mit produktudvalg  
således, at varekataloget altid er up-to-date

Accept kriterier:

Vareliste og priser skal kunne opdateres

Estimat:

Front end: M

Back end: XL

Home -> Log Ind -> Log ind -> Admin -> Vareliste

Fra forsiden trykkes der på Log ind i øverste højre hjørne. Derefter indtastes admin bruger info

Brugernavn: admin

Kodeord: 1234

og der trykkes på log ind.

Herefter trykkes der på admin i navigationsbaren og derefter på Vareliste.

Der vises derefter en liste over alle varer samt priser i produktudvalget.

Trykker man på ret kan man opdatere alt på den enkelte vare



Tasks

#113 Opdater: material, description, quantity, unit, price\_per\_unit

*Som fog medarbejder skal jeg kunne opdatere en vare for at kunne rette i eventuelle fejl*

#128Fog: Fremvisning af pris

*Som fog medarbejde skal jeg kunne se en samlet pris for at kunne redegøre/redigere i den*

#129 SVG view dynamisk af forespørgsel

*Som for medarbejder skal jeg kunne se et billede af en forespørgsel*

**#7 Fog: spec. carport (Redigering)**

*Som Fog-medarbejder*  
Vil jeg kunne redigere i en kundeforespørgsel således, at en kundens rettelser bliver tilføjet

Accept kriterier:

Carportens mål, vinkler og valgmuligheder skal kunne ændres på den individuelle forespørgsel

Estimat:

Front end: L

Back end: XL

How-to:

Home -> Log Ind -> Log ind -> Åben -> Se tegning og stykliste

Fra forsiden trykkes der på Log ind i øverste højre hjørne. Derefter indtastes admin bruger info

Brugernavn: admin

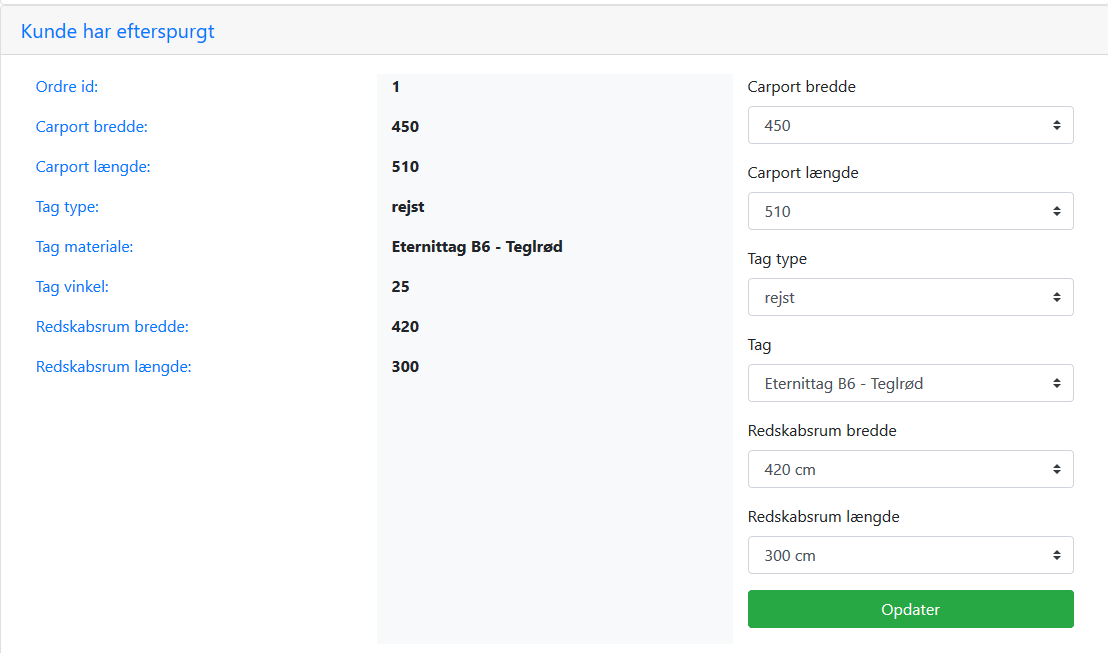
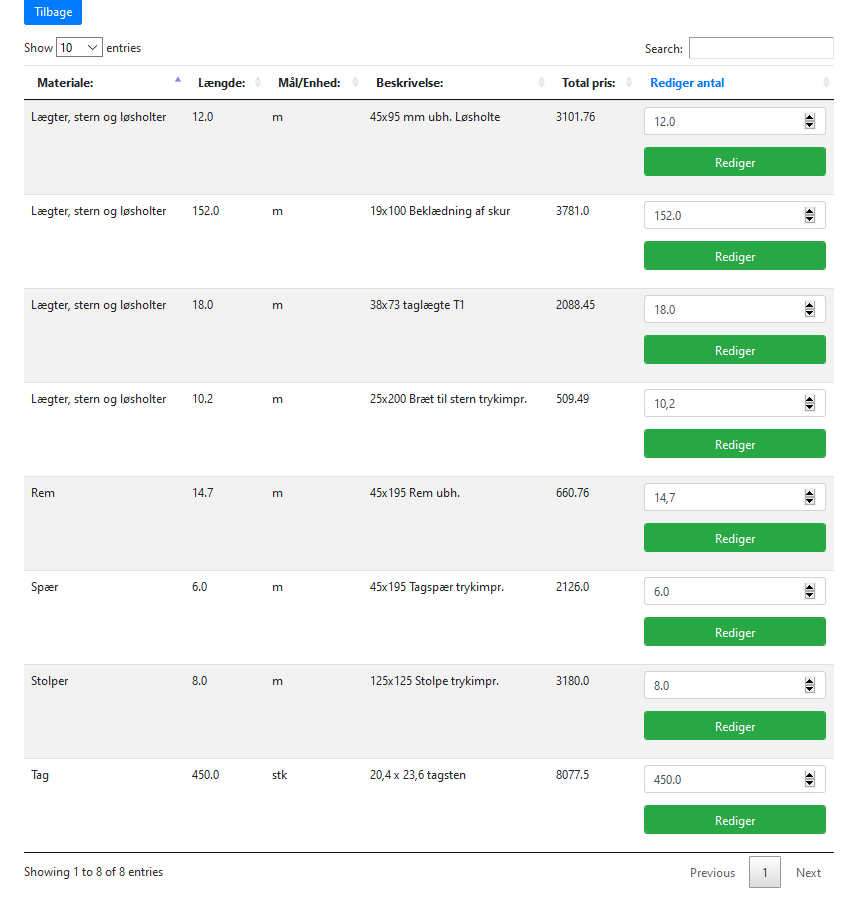
Kodeord: 1234

og der trykkes på log ind.

Herefter trykkes der på ”åben” ud for en forespørgsel i højre side.

På forespørgsel siden kan man derefter opdatere kundens valg af mål, materier samt info.

Derefter kan man nederst på forespørgsel siden trykke på ”se tegning og stykliste”. Hvorefter man også kan opdatere styklisten.

Tasks

#140 Select option værdier

*Dropdown menuer skal have værdier*

#137 Fog: Opdatering af stykliste mål

*Stykliste siden skal kunne redigere i mål og antal af varer*

#114 redigere mål på en forespørgsel

*Forespørgselsiden skal kunne redigere i mål og valg i en forespørgsel og sende dem videre til datamapper*

#138 Fog: Opdatering af stykliste antal fra orderline id

*Databasequery skal laves for at opdatere mål udfra orderline id*

# 6 Diagrammer

## 6.1 ER Diagram

*Det interesante ved denne domæne og database er at den langt hen af vejen er grundlaget for resten at systemet. Tabeller og relationer siger noget om hvad systemet arbejder med, ikke hvordan. Så det er godt sted at starte.*

*Som led i beskrivelsen af ER diagram skal man have følgende med:*

*Diagram over hele modellen. Det er vigtigt at få plads til alle tabeller og alle relationer. Det kan så betyde at man ikke kan få plads til alle attributter på de enkelte tabeller. Dem kan man slå op i databasen, så det er ikke så vigtigt*

*Hvis nogle af tabellerne ikke er på 3. normal form vil det være almindeligt at nævne det, og forklare hvorfor det er gjort (tidspres eller anden overvejelse).*

*Hvis der anvendes 1-1 relationer kan man beskrive hvorfor man ikke blot har en tabel.*

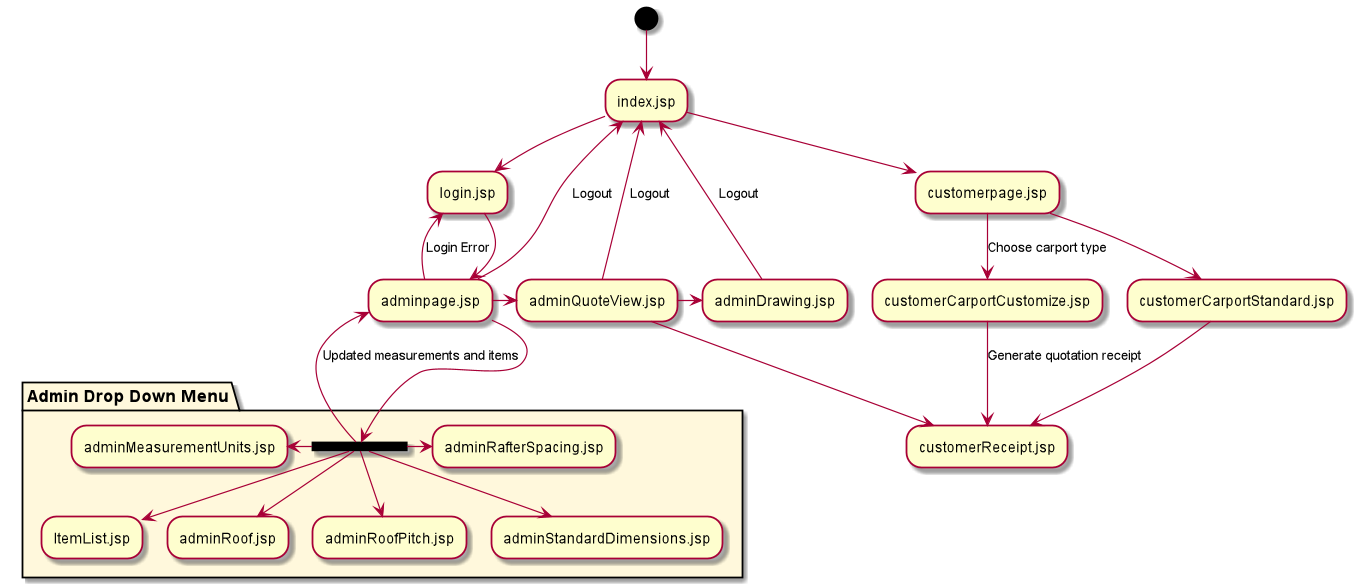
*Hvis nogle tabeller implementerer en mange-mange relation vil det være normalt at beskrive det.*

*Hvis der er flere veje at nå fra et sted til et andet vil det nemt gøre det svært at holde databasen konsistent. Hvis I har gjort det alligevel så skal I forklare hvorfor.*

*Hvis der er tabeller hvor man benytter andet end et automatisk generetet ID som nøgle skal man forklare det.*

*Der er interessant at beskrive hvilke overvejelser der ligger til grund for de konkrete valg der er i ER modellen (fremmednøgler, constraints, triggers, osv)*

## 6.2 Navigationsdiagram



Ovenstående diagram afspejler navigationen gennem Fogs hjemmeside, som enten kunde eller Fog medarbejder. Dette er beskrevet for at få en bedre forståelse af brugeroplevelsen på siden.

Brugeren starter på velkomstsiden index.jsp hvor der vises en navigation bar i toppen af siden. I denne navigation bar er det muligt for brugeren at trykke Home, Bestil eller Log ind. Trykker brugeren Home bliver siden redirected til velkomstsiden igen og fungere på samme måde på samtlige jsp sider.

Er brugeren en kunde har brugeren mulighed for at trykke på Bestil i navigationsbaren og sendes derved hen til customer.jsp hvor der er følgende to valgmuligheder. Den ene er “Bestil - pr. stk. 23.998,-” (customerCarportStandard.jsp) hvor det er tiltænkt at kunden skal kunne bestille carporte på standardmål. Denne mulighed er dog ikke implementeret da product owner ikke ønskede dette på nuværende tidspunkt.

Klikker brugeren derimod på “Byg selv” (customerCarportCustomize) sendes brugeren en til en side hvor det gør det muligt for brugeren at bestille en carport med specielmål samt tagtype- og farve. Efter kunden har tastet alle mål, valg, oplysninger og bemærkninger har kunden mulighed for at trykke på send. Derved sendes kunden hen til en kvitteringsside (customerReceipt.jsp) hvor kunden modtager en kvittering på at deres forespørgsels er gennemført. Denne kvittering kan også printes hvis kunden vil det.

Går vi tilbage til udgangspunktet index.jsp og lader nu en Fog medarbejder logge ind på hjemmesiden, bliver Fog medarbejderen sendt hen til adminpage.jsp hvor der gives en række valgmuligheder. Navigationsbaren ændrer sig og indeholder nu Forespørgsels Oversigt samt en drop down menu. Hvis Fog medarbejderen vil kigge, rette eller slette en bestemt forespørgsels kan han trykke på knapperne “Åben” for at se og rette eller Fjern for at fjerne en forespørgsels.

Trykker Fog medarbejderen på “Åben”, sendes medarbejderen hen til adminQuoteView.jsp hvor der kan rettes og slettes i de informationer kunden har indtastet. Derudover har Fog medarbejderen også mulighed for at trykke på knappen “Se tegning & stykliste”, som viser tegningen af den pågældende carport. Her sendes medarbejderen hen til adminDrawing.jsp hvor tegningerne og stykliste bliver genereret og vist.

Hvis Fog medarbejderen vil se samme kvittering som kunden kan han/hun trykke på “Se tilbud” og bliver dermed sendt til customerReceipt.jsp.

Til sidst har Fog medarbejderen mulighed for at gøre brug af drop down menuen hvor der kan rettes følgende ting: Måleenheder (adminMeasurementUnits.jsp), Vareliste (ItemList.jsp), Rafterafstand (adminRafterSpacing.jsp), Tagbeklædning (adminRoof.jsp), Taghældning (adminRoofPitch) og Standardmål (adminStandardDimensions.jsp).

# 7 Særlige forhold

*Dette afsnit bruges til at beskrive særlige forhold der benyttes i programmet. Det kan f.eks. være:*

*Hvilke informationer gemmes i session*

*Hvordan håndterer man exceptions.*

*Hvordan man på har valgt at lave brugerindput validering*

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

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

*… andre elementer – i Fog projektet kan det være:*

*Tegning*

*Stykliste beregner*

*Husk: det er bedre med 2 linjers dokumentation end ingen.*

# 8 Udvalgte kodeeksempler

*Det er ikke sikkert at censor (eller eksaminator) finder alle jeres guldkorn i selve koden. Derfor er det en god ide at vælge særlige kode stumper ud og vise dem i rapporten.*

*De eksempler der er givet uder “særlige forhold” afsnittet kan man godt tage og illustrere med kode direkte i rapporten.*

*Det kommer til at virke særligt overbevisende hvis den kode man vælger ud indgår som led i et af sekvensdiagrammerne.*

*Der er mange af jer der vil skrive jeres ting i word eller googledocs. Vær opmærksom på hvordan i formaterer jeres kode. Man vælger ofte en lidt mindre font, en der er “monospaced” (alle bogstaver optager samme bredde). Der er også nogle der sætter små skærmbilleder fra Netbeans ind. Det er OK, men så husk at vælge et tema fra netbeans med hvid baggrund og mørke/farvede bogstaver da nogle censorer skriver rapporten ud på blækprintere som ikke gengiver lyse bogstaver på sort baggrund særligt godt.*

# 9 Status på implementation

Da vi er 5 i vores gruppe lagde vi ud med at sætte høje forventninger til os selv og hvad vores system skulle kunne udføre. Det skulle dog vise sig at flere uventede ting dukkede op gennem vores udvikling af projektet.

Lad os starte med vores .jsp side customerCarportStandard, hvor vi havde ønsker om at en kunde skulle kunne bestille en færdig bygget carport med fastlagte mål. Man skulle kunne være i stand til at se en side med tekst og pris til den valgte carport. Herfra skulle man kunne navigere sig videre til en anden side, som skulle give en større og bedre oversigt over vores udvalg, samt bedre beskrivelse af den pågældende carport. Vi formået dog ikke at få henstillet tid til at få taget os af den del. Det var dog ikke en user story, men bestemt noget vi godt kunne have tænkt os, at have haft med ind over vores projekt. Vi havde stor fokus på den dynamiske del ved at kunne bygge sin egen carport og at kunne få genereret arbejdstegninger dynamisk til den pågældende forespørgsel.

Vi kan kort nævne hvad vi kunne have gjort kodemæssigt, da det ikke er det mest komplekse, der skulle til for at komme i mål med denne opgave. Vi skulle starte med at lave input felter, på .jsp siden, for at en kunde kunne indtaste information om personens navn, adresse, telefon, e-mail og en eventuel kommentar til købet. Herfra skulle man fra sin kaldte java klasse sendes til LogicFacade, som sender ens kald videre til DataMapper klassen for til sidst at ramme en metode med en INSERT query, som vi allerede havde lavet til vores byg selv forespørgsler. Den nødvendige information, som skulle sendes videre sendes via metode parameterne, rammer en metode i DataMapper klassen, med fastlagte variabler for den pågældende standardcarport.

Derudover kunne vi godt kunne have tænkt os at have brugt mere tid på den visuelle del af vores hjemmeside. F.eks. kunne der bruges mere tid på vores styling af alle .jsp siderne, samt et par justeringer til det overordnede design af hjemmesiden. Vi kan dog være stolte af at vores hjemmeside, stort set, er bygget til at være responsive. Det vil sige at hjemmesidens elementer tilpasser sig skærm størrelsen. Dette er opnået ved hjælp af frameworket Bootstrap. Ellers ville vi, i vores gruppe mene, at vi kom godt i mål med resten, hvad angår .jsp sidernes visualisering og deres funktioner.

Da tiden løb fra os i sidste ende, fik vi ikke brugt den nødvendig tid på at refactor vores kode. Det har bestemt været en lærings proces, i form af ens kode opsætning og fremgangsmåde. Vi skulle have fået gjort det løbende og ikke tænkt, at vi kunne tage tid til det sidst i processen. Et godt eksempel kunne være vores kalkulations klasse, hvor der er mange elementer som bliver instansieret i constructoren. Det skulle være delt op i metoder, da det ville gøre koden overskuelig og nem tilgængelig.

Vi kunne også have tænkt mere over brugervenligheden på vores landing page(adminpage.jsp), når en bruger/administrator logger ind i systemet. Her menes der f.eks., at man til oversigtstabellen kunne have tilføjet status på diverse forespørgsler. Det vil sige om forespørgslen er godkendt eller afvist, hvilket er en funktion man kan se og redigere på oversigts siden(adminQuoteView.jsp).

Vi har også haft i tankerne, at en administrator skal kunne have mulighed for at oprette en ny medarbejder. Det er der gjort plads til og på adminpage.jsp er koden også gjort klar til brug. Alle funktionerne virker hertil, men vi har valgt at ud kommentere koden, da det ikke har været en del af en user story. Samme stykke kode kunne også være brugt til at oprette kunder til systemet, og blot ændre i ”role” fastlæggelsen, som gemme i databasen, hvis dette skulle blive relevant for Fog i fremtiden.

Get/Post funktionerne har også været til diskussion i vores gruppe og vi har valgt i vores gruppe at holde os til Post i alle vores <form>. Det kan dog diskuteres om SELECT funktioner ikke skulle være Get, men her mangler vi en større viden om brugen af Get/Post. Det har dog ingen effekt på brugen eller brugen af vores system.

Exception kunne vi også have tænkt os af at have brugt mere tid på, da vi kun forholder os til LoginSampleException, SQLException og ClassNotFoundException. Vi har flere gange vendt det internt i gruppen, at vi skulle få udarbejdet vores egne brugerdefinerede exceptions, som ville blive opsat ligesom LoginSampleException. Derved kunne man også lave en brugervenlig besked, ved en fejl, samt gøre bedre brug af log systemet, hvilket leder os til det sidste i denne sektion.

Der var et ønske om at få implementeret brugen af en logger, så vi kunne logge vores fejl i systemet og få segmenteret vores fejl(exceptions). På grund af tidsmangel har det dog ikke været muligt. Undervisningsmateriale omkring dette har været set og er bestemt noget vi vil have i tankerne til næste projekt.

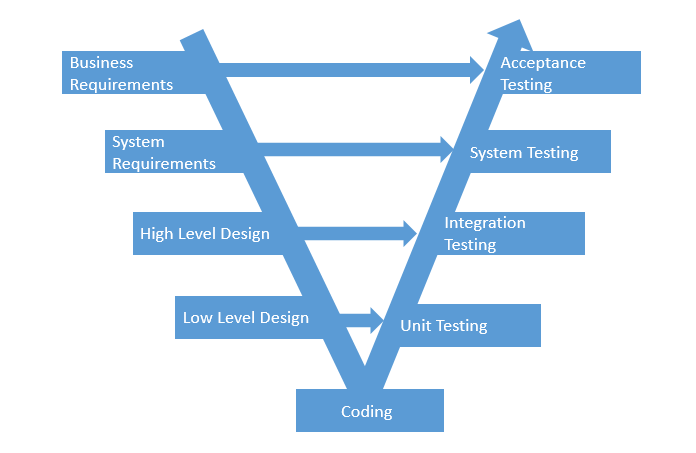
# 10. Test (Husk word fil fra Jon her og indrag v-model)

*Der skal være lavet test. Du kan dokumentere tests ved at beskrive i tabel form:*

*Hvilke klasser er testet*

*Hvilke metoder der er testet*

*Dækningsgrad af dine tests for de valgte metoder og klasser*



*Desuden kan du beskrive hvordan i systematisk har arbejdet med at teste koden før den er blevet gjort til en del af master branch.*

# 11 Process

Der skal være et afsnit hvor I beskriver jeres arbejsprocess i projekt perioden. Der skal dels være et faktuelt afsnit og et reflektions afsnit.

## 11.1 Arbejdsprocessen faktuelt

Dette afsnit skal beskrive:

Hvilke sprints der var, og hvilke user stories der blev arbejdet med.

Hvem der var scrum master i hvilke dele af projekt perioden. Giv gerne nogle eksempler på hvad scrum master gjorde i udvalgte sprint.

Et eksempel på et af PO-møderne, hvad der var planlagt fra jeres side, og hvordan det gik.

Hvordan i afholdt jeres daglige standup møder.

Hvornår i holdt retrospectives.

## 11.2 Arbejdsprocessen reflekteret

Dette afsnit skal beskrive jeres overvejelser over hvilke dele der har fungeret godt og hvilke dele der måske er faldet lidt på gulvet. I kan f.eks. beskrive:

Om scrummaster rollen fungerede, hvilke problemer I så i den, og hvad I gjorde for at rette op på det.

Hvad der var de væsentligste emner på jeres retrospektiv møder

Om I havde problemer med at nedbryde user stories i tasks

Om I var spot-on med jeres estimeringer

Om der var problemer med vejledningen og PO møderne

Hvor langt inde i processen I fandt en rytme der var produktiv

Andre elementer der har at gøre med at forsøge at arbejde i et scrum team

12 Konklusion

En konklusion er en sammenfatning af de resultater, man har fundet i forbindelse med udarbejdelse af rapporten. Dette skal dog ikke forstås som et resumé af, hvad man har skrevet, men som et svar på de spørgsmål man stillede i sin problemformulering. Det er meget vigtigt, at der bliver svaret på de spørgsmål, der blev stillet i problemformuleringen, men konklusionen skal kun indeholde de vigtigste pointer og påstande. Der må ikke inddrages nyt stof i en konklusion.

13 Perspektivering

Ønsker man at inddrage andre perspektiver, kan man lave et afsnit efter konklusionen, der hedder ”Perspektivering”. Dette afsnit er ikke obligatorisk, men kan bruges til at beskrive, hvilket videre arbejde rapporten danner grundlag for, hvilke syn der kunne være interessante at inddrage i andre sammenhænge, hvilke paralleller der kan drages til andre områder, osv. Perspektiveringen er en diskussion af konsekvenserne af konklusionen i et større perspektiv.

# 14 Appendix

## 14.1 User Stories

## Sprint 1

**#21 Kunde: Forespørgsel på spec. carport (basic)**

Som Kunde

Vil jeg kunne forespørge på en carport i specialmål (L,H,B) med standard tag, u. skur

Således at jeg kan modtage et tilbud på samme.

Acceptkriterier:

Kunden skal kunne angive udvalgte mål til længde, højde, bredde.

**Tasks**

#30 Krav til dimensionering skal implementeres (Antal søjler, dimensioner mm)

#54 Kundeforspørgsels <select> skal populeres med korrekt data

*Vælg tag*

*1 - valg tagtype: fladt / rejst*

*2 - hvis fladt = kun ét slags tag*

*3 - hvis rejst = vælg hældning*

*4 - hvis rejst = vælg beklædning*

#39 Design konverteres til kode

#27 Database: E/R Diagram

#51 Beregning af lægter

#48 Database: Queries til tabelopslag på raft length/spacing/dimension

#24 Opsætning af command-pattern skabelon og klargøring på git med MASTER og PRODUCTION branch

#26 Design: Hjemmeside design til index og customerpage

#37 Database og program skal forbindes (Med standard credentials)

#28 Domæne model (.uml)

#29 Opstart på klasse diagram

**#22 Kunde: Forespørgsel på spec. carport (basic m. tag)**

Som kunde

Vil jeg kunne forespørge på en carport i specialmål (L,H,B) med enten rejst eller fladt tag  
Således at jeg kan modtage et tilbud på samme.

Accept kriterier:

Kunden skal kunne vælge tagtype og hældning

**Tasks**

#38 Carport beregninger laves om til metoder

#32 Kunden kan selv vælge tag (Fladt, rejst m. hældning i x grader)

**#23 Kunde: Forespørgsel på spec. carport (basic, tag + skur)**

Som kunde

Vil jeg kunne forespørge på en carport i specialmål (L,H,B) med rejst eller fladt tag inkl. skur i specialmål

Således at jeg kan modtage et tilbud på samme.

Accept kriterier:

Kunden skal kunne angive udvalgte mål til længde, højde, bredde til skur og sende en forespørgsel til fog.

**Tasks**

#47 Design til spec. byg (Customer)

#33 Forespørgsel på skur i kundespecifikke dimensioner

**#34 Kunde: Forespørgsel på materialetyper**

Som kunde

Vil jeg kunne vælge blandt materialetyper og farver  
således at jeg får præcis den carport jeg vil have

Accept kriterier:

Kunden skal kunne vælge de materialer han ønsker

**Tasks**

#36 Hvilke byggematerialer tilbyder fog (Tagtyper, Træsorter, metaller, skruer, beslag mm)

**Storyless tasks**

#45 Design til customer landing page

#43 Layoutside til admin forespørgsel

#44 Fog SVG logo

#42 Opstart på 2D SVG (Learning to walk)

**Sprint 2**

**#5 Fog: Se forespørgsler**

Som fog-medarbejder  
Vil jeg modtage forespørgsler på en carport  
Således, at jeg kan se forespørgslen på hjemmesiden

Accept kriterier

Forespørgsel inkl. beregning (Mål - ikke pris) skal være vist

**Tasks**

#70 Database: Stumpe liste

#78 Main mapper klasse

*Efter vejledningen fra Arne fik vi af vide, at der kun skal bruges en mapper klasse og ikke flere*

#55 Forespørgselsside – designoplæg

#77 Forespørgselsside – jsp

**#15 Fog: Generering af tegning**

Som fog-medarbejder

Vil jeg kunne generere en tegning på kundens forespørgsel  
således, at jeg kan visualisere løsningen for kunden

Accept kriterier:

Dynamiske tegninger af kundens forespørgsler skal generes og være vist

**Tasks**

#66 SVG tegning forfra

#64 SVG oppe fra

#63 SVG mål

#67 SVG tegning fra siden

#69 xd design til SVG landing page

#56 Hente data fra database (dimensioner tabel) til program

**Sprint 3**

**#80 Dynamisk view a SVG**

Som fog medarbejder skal jeg kunne se en dynamisk tegning af en forespørgsel

Accept kriterier

Dynamiske tegninger af kundens forespørgsler skal generes og være vist

**Tasks**

#91 Hente parameter fra DB

#95 Klasse diagram opdateres

**#82 Fog: Se forespørgsel**

Som en fogmedarbejder vil jeg kunne se en kundes forespørgsel således at jeg kan starte på et tilbud

Accept kriterier

Kunders forespørgsler skal kunne vises

**Tasks**

#102 Validering af select options

#96 SVG admin view jsp side laves

#93 Kunde forespørgsel adresse info gemme i db

#107 Åbne kunde quote dynamisk

#99 Quote overview redesign

#94 Kunde forespørgsel (mål) gemmes i db

#97 Admin overview over alle forespørgelser

#108 request info fra specifik kunde til overview.jsp

**#8 Fog: Slette forespørgsler**

Som fog-medarbejder

vil jeg kunne slette en forespørgsel således at fejlkøb, fortrydelser mm. ikke er i systemet

Accept kriterier

Forespørgsler skal kunne slettes samt være slettet i databasen

**Tasks**

#89 Form opsætning

#76 Query til DB

**#14 Validering af spec. løsninger**

Som fog-medarbejder

vil jeg hjælpes af systemet til at vurdere om kundens løsning er realistisk  
således at vi sælger en løsning der kan lade sig gøre

Accept kriterier:

Mål og valgmuligheder skal valideres således at en carport kan realiseres udfra en forespørgsel

**Tasks**

#106 Validering på bestillingssiden

**#109 Kunde: Kvittering ved forespørgsel**

Som kunde

Vil jeg modtage en kvittering når jeg sender en forespørgsel afsted til fog

Accept kriterier:

Kvittering med de valgte mål, materialer og valgmuliger skal vises efter afstendt forespørgsel

#111 Vis kvittering

**Storyless tasks**

#104 Spær rettes til

#103 Opret getters til carportcalc

#105 Databasetabel til ”assumption” dimensioner

**Sprint issues**

#100 eternit tag på rejst tag skal fjernes

*Man skal ikke kunne vælge eternittag på rejst tag?*

**Sprint 4**

**#11 Fog: Administrering af varekatalog**

Som fog-medarbejder  
vil jeg kunne opdatere varer og priser i mit produktudvalg  
således, at varekataloget altid er up-to-date

Accept kriterier:

Vareliste og priser skal kunne opdateres

**Tasks**

#113 Opdater: material, description, quantity, unit, price\_per\_unit

*Som fog medarbejder skal jeg kunne opdatere en målinger for at kunne rette i eventuelle fejl*

#128Fog: Fremvisning af pris

*Som fog medarbejde skal jeg kunne se en samlet pris for at kunne redegøre/redigere i den*

#129 SVG view dynamisk af forespørgsel

*Som for medarbejder skal jeg kunne se et billede af en* forespørgsel

**#7 Fog: spec. carport (Redigering)**

Som fog-medarbejder  
vil jeg kunne redigere i en kundeforespørgsel  
således, at kundens rettelser bliver tilføjet

Accept kriterier:

Mål, vinkler og valgmuligheder skal kunne ændres på den individuelle forespørgsel

**Tasks**

#140 Select option værdier

#137 Fog: Opdatering af stykliste mål

#114 redigere mål på en forespørgsel

#138 Fog: Opdatering af stykliste antal fra orderline id

**Storyless tasks**

#133 Design clean up

#134 Stykliste fremvisning

#136 Exception clean up i DB

#135 Code: redesign code til at søge på order id og ikke user id

#124 readme fil på github

#110 opdater plantUML diagram(er)

#123 SVG tegning forfra

#119 SVG toString()

#121 SVG fra siden: teglsten

#122 2 versioner af tegning fra siden

**Sprint issue:**

#125 metoder skal hentes statisk og ikke som objekter

**Sprint 5**

**#171 Slutspurt**

Få færdiggjort de sidste ting inden deadline

Accept kriterier:

Skal være lukket inden fredag d.29/5

**Tasks**

#190 CSS oprydning/style

#186 jUnit skal laves på udvalgte klasser/metoder

#193 N2H: SVG crasher når man går fra rejst til fladt tag

#187 JavaDoc kommentarer på alt

#191 Lorem ipsum skal rettes

#194 ”Se tilbud” funktion

#197 form group button fix

#189 fladt tag og u. Skur skal init. Til 0 for at undgå null-pointer

#184 løsdelspakker skal tilføjes til priskalkulationen

#183 pris skal i database inden den vises på kundeforespørgselssiden

#188 ”Ryd op efter dig selv”- klasser og testmetoder

**#6 Fog: spec. carporte (Modtaget tilbud)**

Som Fog-medarbejder  
Vil jeg kunne godkende et tilbud på spec. carporte  
således, at jeg kan fremsende ordrebekræftelse, stykliste og byggevejledning

Accept kriterier:

En kundens forespørgsel skal kunne godkendes, og der skal vises en ordrebekræftelse, stykliste samt byggetegninger.

**Tasks**

#199 antal og pris hænger ikke sammen

#195 SVG fixes

#198 fravælge redskabsrum

#196 item list crash

#130 godkend quote

#144 dropdown til status på kundeforespørgsel

**#148 Fog: CRUD**

Som Fog-medarbejder  
Vil jeg kunne rette, slette og opdatere priser og enheder i databasen  
Således at systemet er langtidsholdbart og fleksibelt

Accept kriterier:

Der skal kunne rettes, slettes og opdateres priser og enheder i databasen fra hjemmesiden af.

#149 Design til ”måleenheder”

#154 klasse & funktionalitet til ”måleenheder”

#158 klasse & funktionalitet til ”standardmål”

#156 klasse & funktionalitet til ”raftermål”

#169 klasse & funktionalitet til ”taghældning”

#155 klasse & funktionalitet til ”vareliste”

#157 klasse & funktionalitet til ”tagbeklædning

#159 Design til ”måleenheder”

#150 design til ”vareliste”

#152 Design til ”tagbeklædning”

#153 design til ”standardmål”

#166 klasse & funktionalitet til ”rafterafstand

#165 klasse & funktionalitet til ”vareliste”

#151 design til ”rafterafstand”

**#59 Fog: beregning af pris**

Som Fog-medarbejder  
vil jeg have at systemet beregner en pris  
således at jeg kan klargøre et tilbud til kunden

Accept kriterier:

Systemet skal kunne beregne en pris på et tilbud til kunden

#60 komplet varekatalog skal udarbejdes(inkl tilbehør)

#86 unit test

#118 dækningsgrad JSP

#112 totalpris ned i DB

#132 tabel til prisforslag

#101 oprette en ”løsdelspakke” til skruer/beslag afhængig af m/u skur og m/u rejst tag

#73 klasse til udregning af pris

#79 carport calculation

#141 priser i DB skal rettes så de er realistiske

#117 dækningsgrad DB

#61 alle varer skal prissættes

#88 datamapper til priser

#81 data fra .jsp selects skal gemmes i db(quotes)

#98 validering på skur dimensioner

**Storyless tasks**

#203 rapport skabelon

#205 svg dokumentation af hver pil

#202 billeder til index carousel

#201 mangler underdeling af packages

#204 svg mål skal være ens mellem alexander og morten

**Sprint issues**

#143 Tag og skur skal sættes til ’0’ i carportcalculation(forhindre null)

#142 raftdistance from db results in “out of bounds” error