

# Carport

27/11/23 - 03/01/24

Patrick Fabrin, [cph-pf73@cphbusiness.dk](mailto:cph-pf73@cphbusiness.dk), Odiegenx  
Christian Høj, [cph-ch633@cphbusiness.dk](mailto:cph-ch633@cphbusiness.dk), chris4567890  
Nicolai Theis Rolin, [cph-nr140@cphbusiness.dk](mailto:cph-nr140@cphbusiness.dk), Sir Rolin  
Nicklas Waldemar Seier Winther, [cph-nw89@cphbusiness.dk](mailto:cph-nw89@cphbusiness.dk), NokIkNick

2022E2Sem - Gruppe B  
(Færdigskrevet d. 02/01/24)

# Indholdsfortegnelse

<b>Indholdsfortegnelse</b>	<b>1</b>
<b>Links:</b>	<b>2</b>
Github repository: <a href="https://github.com/SirRolin/Carport---GroupB">https://github.com/SirRolin/Carport---GroupB</a>	2
Link til demo video: <a href="https://www.youtube.com/watch?v=Lwt5gP2HhA">https://www.youtube.com/watch?v=Lwt5gP2HhA</a>	2
Link til hjemmeside: <a href="http://159.223.24.167:7070">http://159.223.24.167:7070</a>	2
<b>Indledning:</b>	<b>2</b>
<b>Baggrund:</b>	<b>3</b>
<b>Virksomheden/Forretningsforståelse:</b>	<b>3</b>
<b>Teknologivalg:</b>	<b>5</b>
<b>Krav:</b>	<b>6</b>
<b>Fremvisning af bestillinger:</b>	<b>12</b>
<b>User stories:</b>	<b>13</b>
<b>Domæne model og ER diagram:</b>	<b>16</b>
<b>ER-diagram:</b>	<b>17</b>
<b>Navigationsdiagram:</b>	<b>18</b>
<b>Mockups:</b>	<b>19</b>
<b>Valg af arkitektur:</b>	<b>20</b>
<b>Særlige forhold samt Udvalgte kodeeksempler:</b>	<b>21</b>
Hvordan vi har valgt at håndtere fejl / Exceptions:	25
Hvordan vi har kørt validering på brugerinput:	26
Hvordan vi har valgt at lave sikkerhed i forbindelse med login:	28
Hvilke brugertyper (roller) vi har valgt i databasen, og hvordan de er brugt i jdbc:	29
<b>Status på implementering:</b>	<b>29</b>
<b>Kvalitetssikring (Test):</b>	<b>34</b>
Automatiserede tests:	34
User Acceptance tests:	36
Proces:	38
Arbejdsprocessen faktuelt:	39
<b>Arbejdsprocessen reflekteret:</b>	<b>39</b>
<b>Bilag:</b>	<b>41</b>
Figur: 1	41
<b>Interessentanalyse:</b>	<b>41</b>
<b>User Stories &amp; Acceptance Criteria</b>	<b>44</b>

## Links:

Github repository: <https://github.com/SirRolin/Carport---GroupB>

Link til demo video: <https://www.youtube.com/watch?v=Lwtt5gP2HhA>

Link til hjemmeside: <http://159.223.24.167:7070>

Admin brugernavn	Admin kodeord
admin	password

## Indledning:

Et IT-system med manglende funktioner kan skabe en række problemer hos et firma. Dette kan være alt fra skærpet effektivitet ved brug af systemet til forældet lagerstatus for materialer. Gruppen har derfor fået til opgave at hjælpe firmaet “Johannes Fog” (fremover nævnt som “Fog”) med et system, som øger deres sælgers effektivitet samt gør det nemmere for deres potentielle kunder at bestille deres ønskede produkter. Over projektperioden er der blevet udarbejdet en potentiel løsning til “Fog” i form af en samlet hjemmeside baseret på en undersøgelse og analyse af firmaets behov, ønsker og udfordringer.

Til at løse opgaven fik vi en video med “Fog”s salgschef, Martin, der forklarer deres nuværende system og dets mangler. En af de ting, han lægger meget vægt på, er, at kunden først får adgang til styklisten til carporten efter, der er blevet betalt for ordren. Alt andet information er okay for kunden at få. Det, som kunder betaler for, er servicen i at få en totalpakke, hvor de kan gå ned på “Fog”s lager og få alt det, som de skal bruge til at lave deres nye carport.

Martin kommer ind på, hvordan de før har brugt penge på et nyt system, men endte med at gå tilbage til deres gamle, fordi det nye system ikke fungerede med det lagersystem, de brugte. Dette havde vi i mente, da vi startede projektet. Dette blev svært at følge op på, da det ikke var muligt for os at finde ud af, hvilket system de anvendte (Dette bliver yderligere forklaret i afsnittet “Virksomhedens/Forretningsforståelse”).

## Baggrund:

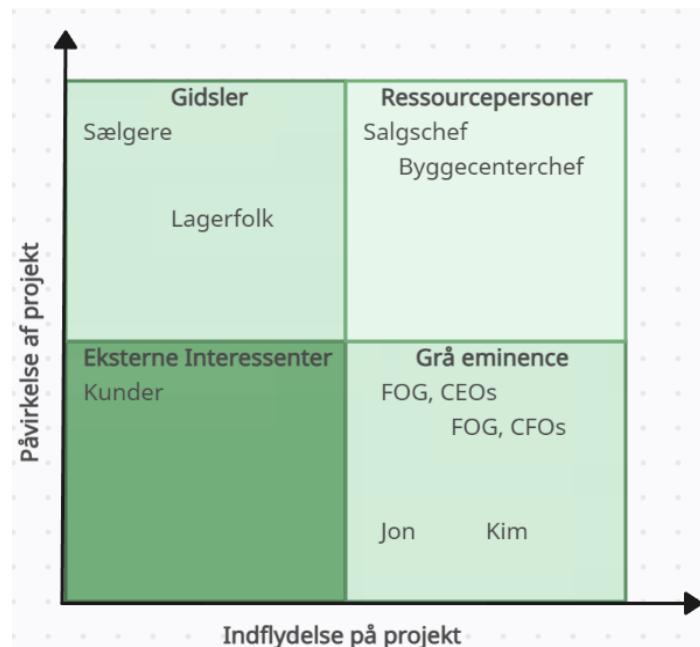
Virksomheden ønsker en hjemmeside, som skulle gøre det nemmere for både sælgere og kunder. De ønsker en hjemmeside, som samler alle de funktioner, som de bruger til at håndtere en kunde ét sted. Det vil sige, at de gerne vil fjerne al afhængighed af eksterne software. Virksomheden vil gerne have, at en kunde skal kunne gå igennem "Fog's" sortiment af varer, udvælge hvad de skal bruge og bestille det eller kunne lave en brugerdefineret bestilling og sende den videre til en sælger. En sælger skal kunne modtage bestillingen, behandle den og derefter kontakte kunden igen.

## Virksomheden/Forretningsforståelse:

Når man arbejder sammen med et firma eller eksterne ressourcer, kan det være relevant at lave en interessenanalyse. Interessenanalysen kan hjælpe en med at identificere de relevante personer til ens projekt, deres vigtighed for projektet og dets resultat. Måden, vi har gjort brug af interessenanalysen på, er, at den hjalp os med at identificere, hvilke interesser vi ville have behov for at involvere i projektet og deres potentielle påvirkning på det.

Som nævnt i indledningen var det et problem med at det nye system "Fog" fik lavet, ikke virkede med deres lagersystem. Derfor endte det med at blive droppet til fordel for at bruge det gamle. Her har vi identificeret, hvem der skulle involveres i vores projekt for at undgå at det samme skete igen (Lagerfolk). Hvis vi havde haft mulighed for at involvere lageret, var en person fra lageret blevet til en ressourceperson for at undgå den samme problematik igen.

Nedenunder kan der findes en model, hvor vi har identificeret og vurderet en række personer baseret på deres påvirkning af projektet og deres indflydelse, samt en tabel med alle personer, hvad de kan opleve af fordele ved projektet, ulemper ved projektet, deres position/bidrag, og hvordan vi vil håndtere de enkelte personer.



Carport			Udfyldt af: Samlet gruppe	Dato: 21/11/23
Interessent	Interessenten kan opleve følgende fordele ved projektet	Interessenten kan opleve følgende ulemper ved projektet	Samlet vurdering af interessentens bidrag/position	Håndtering af interessenten
Kunde	Hurtigere betjening. Færre indtastningsfejl.	Migrations-forstyrrelse.	Lav interesse og indflydelse.	Dem på mail-listen får tilsendt en mail med opdatering. Dem på mail-listen kan blive udvalgt til en demo-hjemmeside, hvor de kan give feedback.
Salgschef	Øget effektivitet for sælgere. Øget kundevenlighed. Højere omsætning i form af flere kunder. Nem opdatering af lager/priser.	Genoplæring af sælgere. Implementering kan tage tid.	Høj interesse og høj indflydelse. Skal medtages i projektet.	Skal informeres og opdateres løbende om projektet. Skal involveres i store beslutninger. Dette gøres ved biweekly møde.
Sælgere	Øget effektivitet på arbejdet. Overskuelighed. Færre indtastningsfejl. Mere tid til kundeinteraktion.	Genoplæring og migrations-forstyrrelser.	Høj påvirkning af projektet, med lav indflydelse. Skal informeres.	Skal informeres om store eller pludselige ændringer. Muligvis med en e-mail eller orienteringsmøde.
CEO	Højere effektivitet for firmaet. Forhøjet aktieværdi.	Periode af tab af penge på produktion.	Lav påvirkning, meget høj indflydelse. Kan stoppe projektet når som helst. Skal høres.	Skal konstant høres og deres beslutning er lov. Dette gøres ved at samle manglende beslutninger og muligt holde et møde ugentligt.
CFO	Profit over tid. Forhøjet aktieværdi.	Kost som bliver genvundet af sælger effektivitet.	Lav påvirkning, meget høj indflydelse. Kan stoppe projektet når som helst. Skal høres.	Skal konstant høres og deres beslutning er lov. Dette gøres ved at samle manglende beslutninger og muligt holde et møde ugentligt.
Bygge Centerchef	Øget effektivitet for byggecentret. Øget kundevenlighed. Øget produktion. Nem opdatering af lager og nye varer.	Genoplæring af arbejdere. Migrationsperiode.	Høj interesse og høj indflydelse. Skal medtages i projektet.	Skal informeres og opdateres løbende om projektet. Og involveres i store beslutninger. Dette gøres ved at tage dem med til det ugentlige møde.

Lagerfolk	Kontinuerligt opdateret data, som passer ind med lagersystemet.	Skal interviewes. Implementering kan tage tid.	Høj påvirkning, men lav indflydelse.	Skal høres om integration med lagersystemet. Dette gøres ved et møde tidligt i processen, et senere i projektet ved behov og ugentligt nyhedsbrev, hvor de kan indsende feedback.
-----------	---	--	--------------------------------------	---

## Teknologivalg:

Teknologi:	Funktioner:
<b>IntelliJ 2023.2.4 (Ultimate Edition).</b>	<b>IDE (Integreret udviklingsmiljø).</b>
<b>Github Desktop 3.3.5 (x64).</b>	<b>Github-interface. Version Control.</b>
<b>Figma</b>	<b>Mockup designer.</b>
<b>Postgresql 15.4.</b>	<b>Database.</b>
<b>pgAdmin 4 7.5.</b>	<b>Database Manager.</b>
<b>Javalin 5.6.1.</b>	<b>Java Web Framework.</b>
<b>Thymeleaf 3.1.1.</b>	<b>Java Template Engine Framework</b>
<b>Junit Jupiter 5.8.2.</b>	<b>Unit Testing.</b>
<b>HikariCP 5.0.1</b>	<b>Java Database Controller.</b>
<b>Java JDK 17.0.9.</b>	<b>Programmeringssprog</b>
<b>Docker Desktop 4.24.2.</b>	<b>Containerization Software</b>
<b>DigitalOcean</b>	<b>Cloud Service Provider</b>
<b>Ubuntu 22.04</b>	<b>Server Styresystem</b>
<b>Docker 23.0.6 for Ubuntu 22.04</b>	<b>Containerization Software til server</b>

## Krav:

Til denne opgave har vi haft et interview med kunden, "Fog", gennem salgschefen Martin, som har givet os et indblik i, hvordan et bestillingsforløb og ordrehåndtering fungerer for firmaet. Således har vi modtaget både deciderede krav til produktet men også 'nice-to-have' krav.

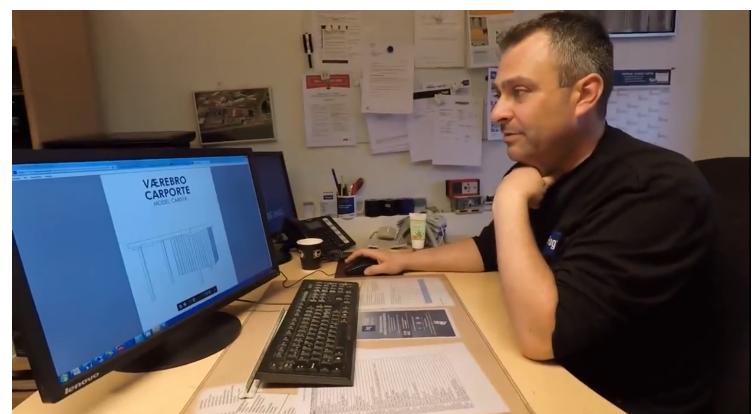
Et af de største ønsker kom i form af, at det ønskes, at hele processen kunne foregå på samme hjemmeside eller program. Det har vi valgt at fokusere på ved at lave et produkt, som samler alle de ønskede funktioner i ét.

Nogle af de vigtigste funktioner lyder således:

- Kunden skal kunne se sin ordre.
- Kunden skal kunne søge på varer.
- Kunden skal kunne se ændringer på deres ordre, så de kan følge med i processen.
- Kunden skal kunne bestille en carport med egne mål.
- Kunden skal kunne se en liste over materialer som carporten består af efter, de har betalt.
- Kunden skal kunne angive, om de selv vil opsætte deres carport eller ej.
- Sælgeren skal kunne danne en plantegning over carporten.
- Sælgeren skal kunne redigere/tilføje/fjerne produkter, så hjemmesiden kan holdes opdateret.
- Sælgeren skal kunne se en stykliste med al information om bestillingen.

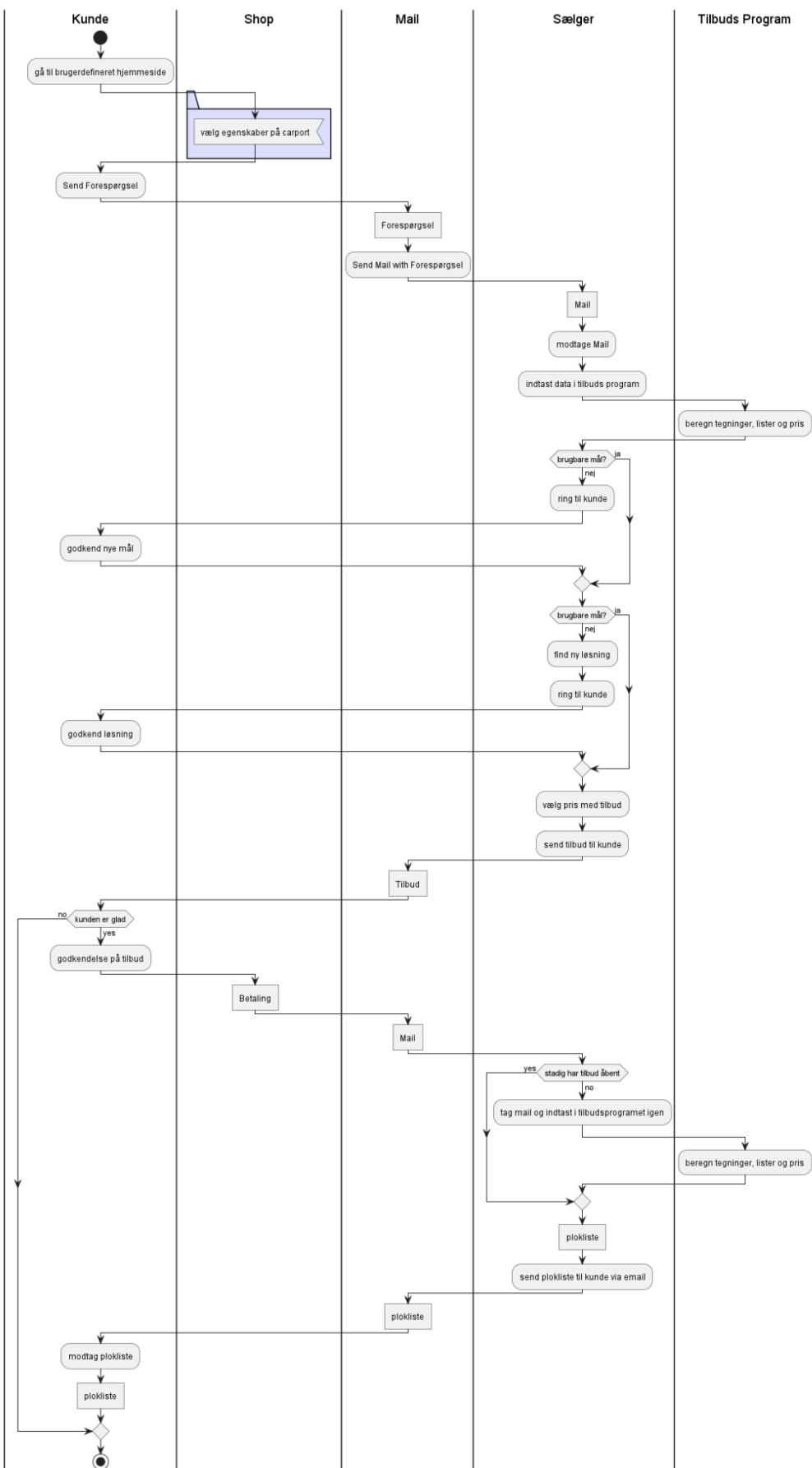
På baggrund af disse ønsker, har vi opsat en række user-stories, som afspejler de funktioner, som vi gerne vil have implementeret. Her har vi også tilføjet ekstra "nice-to-have" user-stories, baseret på både "Fog's" ønsker, samt funktioner vi mente var relevante.

Se afsnit kaldet "User Stories" for mere omkring dette.

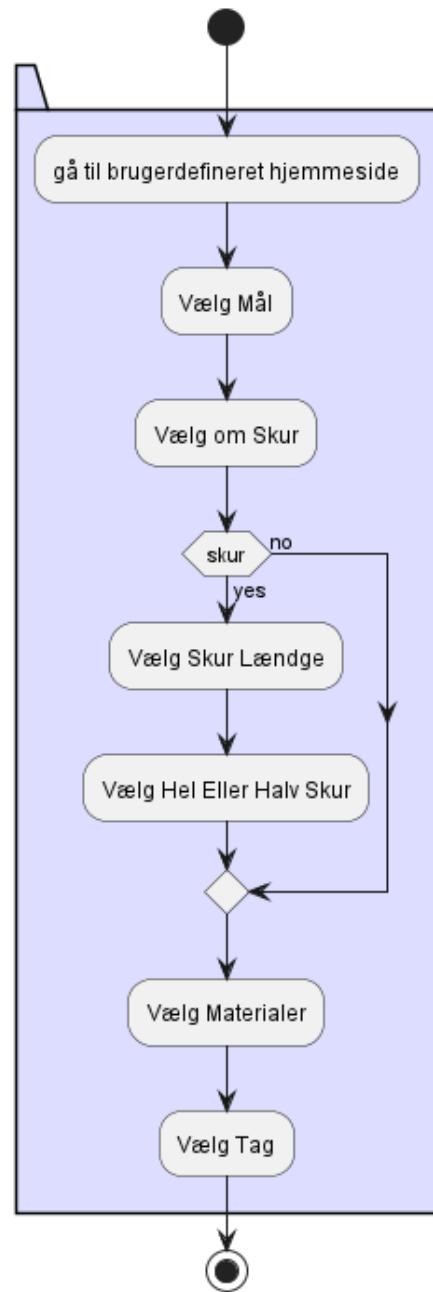


På baggrund af interviewet, har vi også opsat en række aktivitetsdiagrammer, som afspejler de user-stories, der er blevet dannet. Der er både blevet lavet AS - IS diagrammer, som viser, hvordan firmaets processer foregår nu, og TO - BE som viser, hvordan vi ønsker, vores løsning skal foregå.

## AS - IS Aktivitetsdiagram:



Ovenstående aktivitetsdiagram viser, hvordan arbejdsprocessen hos "Fog" er lige nu. De bruger flere forskellige programmer til at løse deres ordrer, som vi gerne vil samle til et. Diagrammet nedenfor hører til AS-IS aktivitetsdiagrammet på forrige side (hvor farven matcher) og beskriver hvordan en kunde vil bestille en custom carport:

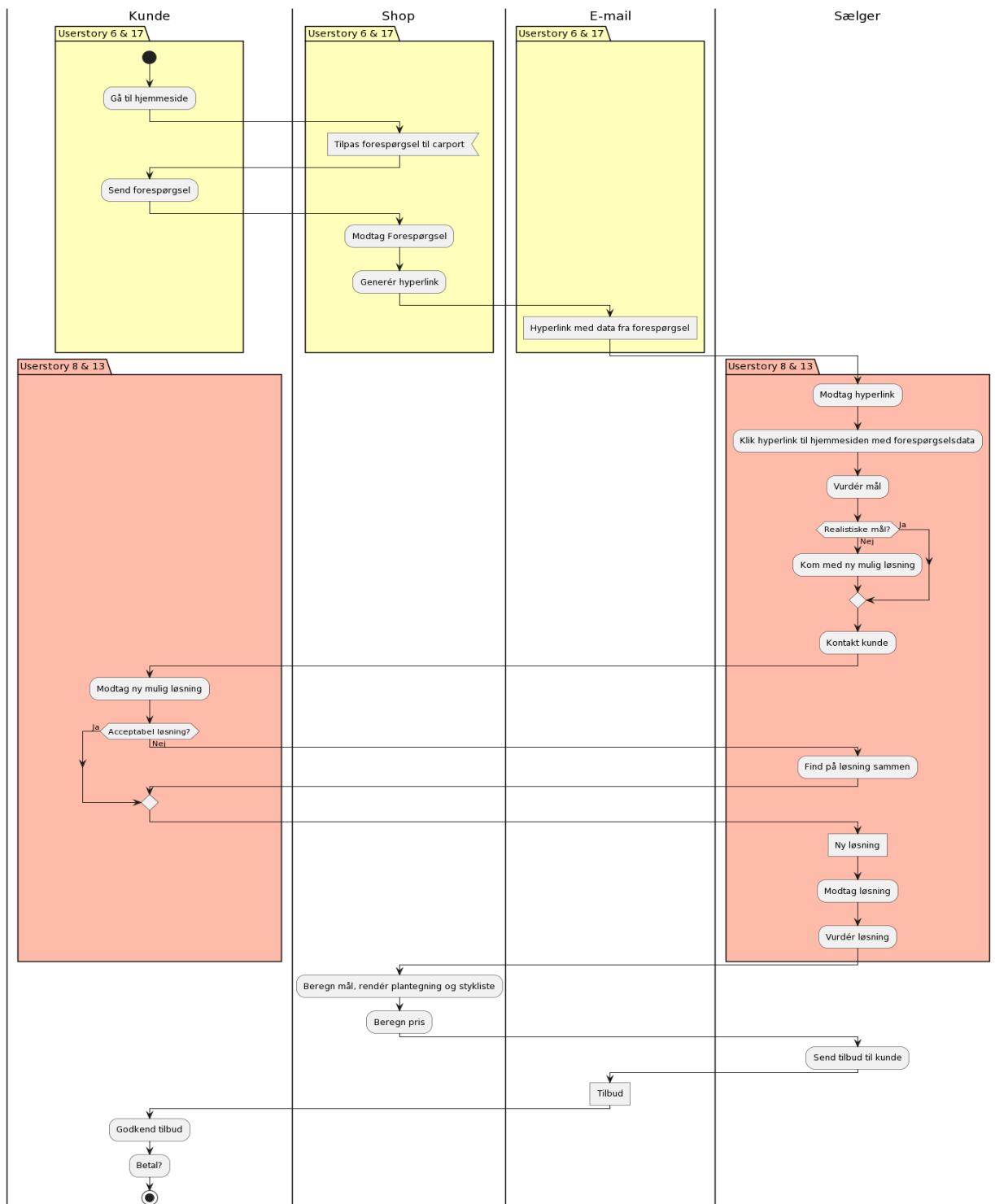


## TO - BE Aktivitetsdiagram:

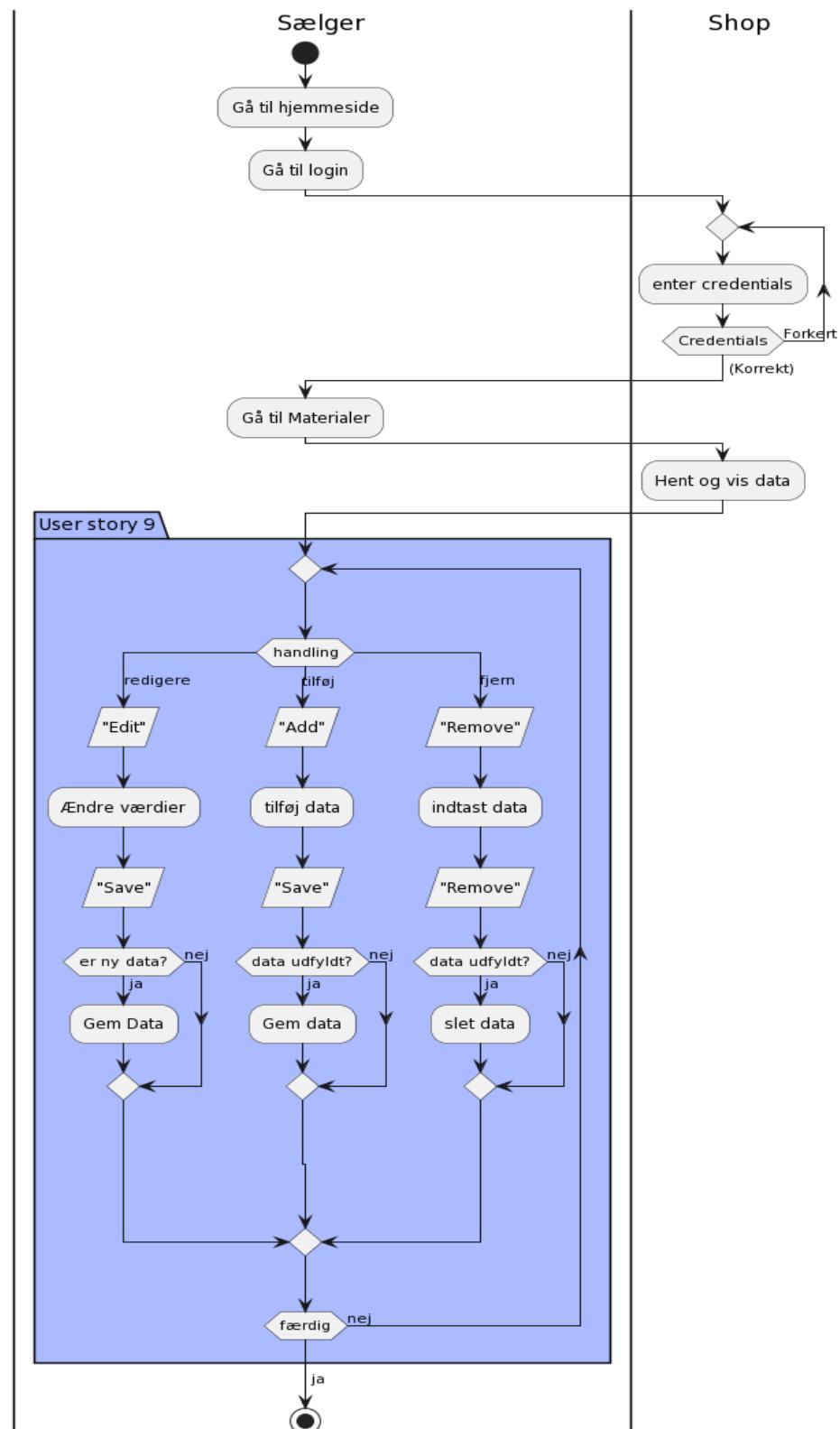
*Bestilling af brugerdefineret carport:*

På denne hjemmeside har du følgende muligheder:

- som admin(sælger) kan man logge ind og ændre i kundens ordrer samt tilføje/fjerne materialer.
- som kunde kan man vælge, om man vil have en carport, der har en incline eller ej og derfra specificere mål og til slut tilføje kontaktoplysninger, før man returneres til startsiden.



*Ændringer af materialer:*



Når du logger ind som sælger(admin), har du 3 muligheder du kan gøre.

1) redigere:

Tryk på knappen edit og herfra muligheden for ændring af data på et eksisterende objekt.

Hvis det er nyt data til det eksisterende objekt gemmer den det og går tilbage til admin siden ellers tilbage til admin siden

2) add:

Giver dig mulighed for at inputte et helt nyt objekt. Skrive navn, pris og længde/højde. Gemme data og sende dig tilbage til admin siden.

3) remove:

Giver dig mulighed for at vælge noget data men i stedet for at inputte noget nyt fjerner den det valgte objekt fra databasen og derefter sender dig tilbage til admin.

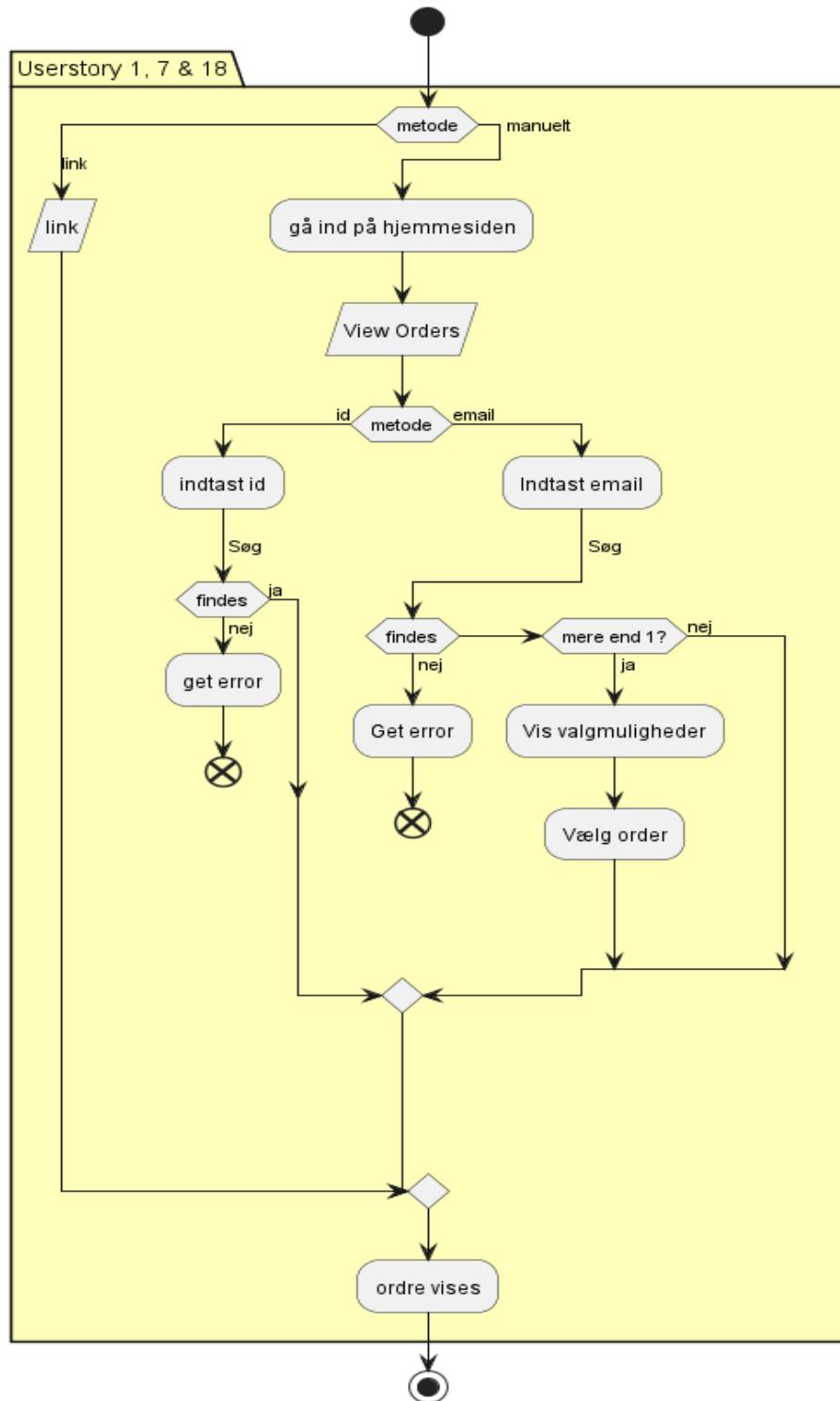
*Fremvisning af bestillinger:*

1) link:

Har du et givet link til din ordre, kan du tilegne dig den direkte ved at taste den ind.

2) søg:

Hvis du ikke har et link, kan du søge på den med ordre-id eller med din email, hvor dine valgte order vil blive fremvist.



## User stories:

Den fulde samling af user-stories kan findes under afsnittet “Status for implementering”, men er også vedlagt som bilag.

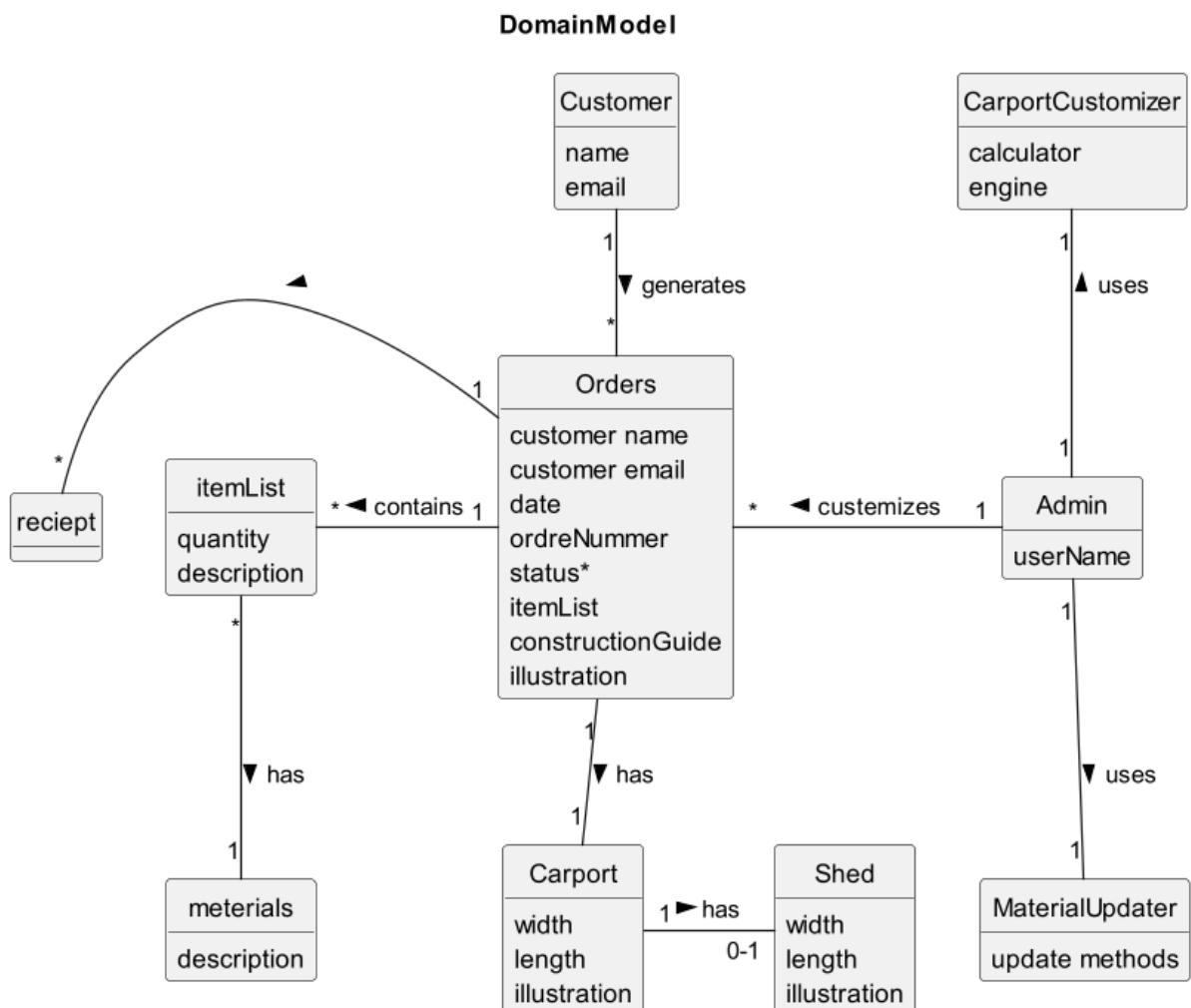
User story:	Tasks:	Accept-kriterie:	Estimat:
<b>1.</b> Som kunde vil jeg gerne se mine ordrer, så jeg kan være sikker på, at jeg har bestilt. (must have)	Lav html side til receipt med tilsvarende controller.  Lav mapper for at hente ordrer.  Lav css til siden.	Givet, at jeg er en kunde, og jeg har bestilt en ordre, så vil jeg gå ind på hjemmesiden og trykke på “view your orders” og så indsætte min order id eller email, og trykke “submit” så jeg kan se statussen på min ordre.	Medium
<b>8.</b> Som sælger vil jeg gerne kunne hente en plantegning over carporten, så jeg kan få en idé om størrelsesforholdet. (must have)	Lav html side med tilhørende controller, hvor man kan generere en plantegning.  Lav en engine der kan beregne og tegne en plantegning.  Lav en mapper så både controllerne og engine kan få den data de har brug for.  Lav css til siden.	Fuldt implementeret. Sælger kan generere plantegningen ved at trykke på en knap og vise den på hjemmesiden og gemme svg teksten, som bliver brugt til at vise billedet, til ordren.	Large

<p><b>6.</b> Som kunde vil jeg gerne kunne bestille en carport efter mine egne mål, for at finde den, som passer mig. (must have)</p>	<p>Lav html side til valg af tag.</p> <p>Lav html side til indsamling af ønskede mål til carporten</p> <p>Lav html side til visning af receipt.</p> <p>Lav tilsvarende CSS</p> <p>Lav controller som tager bestilling og sender den mellem de forskellige sider.</p> <p>Lav mapper der skriver ordren ind i databasen.</p>	<p>Givet, at jeg er kunde, og jeg sidder på "Tilpas carport"-siden, vil jeg kunne tilpasse egne mål og materialer, og derefter lave en bestilling, ved at trykke på "Bestil"-knappen, som derefter sender min bestilling videre til "Fog".</p>	<p>Medium</p>
<p><b>7.</b> Som kunde vil jeg gerne kunne se en liste over materialer som carporten består af, efter jeg har betalt. (must have efter us8)</p>	<p>Lav html side hvor kunden kan se sine ordre, og tilhørende controller.</p> <p>Lav en mapper til at hente information som skal vises på siden.</p> <p>Sikre sig at styklisten kun bliver vist hvis ordren er betalt.</p> <p>Lav tilhørende CSS.</p>	<p>Givet, at jeg er kunde, og jeg har lavet en bestilling, som jeg har betalt for og betalingen er gået igennem hos "Fog", vil jeg kunne trykke på et link, jeg har modtaget via mail, som sender mig ind på en hjemmeside med en kvittering og en liste over materialer, som min bestilling består af.</p>	<p>Medium (såfremt us8 ikke er lavet, ellers small)</p>

<p><b>9.</b> Som sælger vil jeg gerne kunne redigere/tilføje/fjerne produkter, så jeg kan holde hjemmesiden opdateret.(nice to have)</p>	<p>Lav html side til valg af admin funktioner. Lav html side til redigering af materialer og varianter. Lav html til admin login. Lav en controller, som kan vise alle materialer og varianter, kan redigere og tilføje/slette materialer og varianter. Lav mapper til admin siden, der henter data fra databasen og skriver ned i den, og som kan logge ind. Lav tilsvarende CSS.</p>	<p>Givet, at jeg er logget på som admin, og jeg sidder på edit material admin siden, når jeg vælger et produkt, vil jeg gerne have muligheden for at kunne redigere det fra pris til beskrivelse. Givet, at jeg er admin, vil jeg gerne kunne slette det valgte produkt eller tilføje et nyt et.</p>	<p>Large</p>
<p><b>13.</b> Som sælger, skal jeg kunne se en stykliste med al information om bestillingen, så jeg ved hvilke dele, der skal bruges, med deres varenummer, antal, hjælpetekst, beskrivelse, og mål.(must have)</p>	<p>Lav html til visning af styklisten. Lav tilsvarende CSS. Lav controller som kan vise styklisten. Lav mapper som henter styklisten fra databasen og skriver den ind i databasen.</p>	<p>Givet, at jeg har betalt for en brugerdefineret carport, vil jeg gerne kunne hente en plan til byggelsen af carporten, hvis jeg har betalt.</p>	<p>Medium</p>
<p><b>17.</b> Som kunde vil jeg gerne kunne vælge, om jeg hyrer "Fog" til at sætte carporten op for mig eller gør det selv.(must have)</p>	<p>Lav html til siden, hvor man kan se sin receipt. Lav tilsvarende css.</p>	<p>Givet, at jeg som kunde har valgt eller designet min egen carport, vil jeg gerne have muligheden for at kunne bede "Fog" om at sætte den op for mig eller ej.</p>	<p>Small</p>

18. Som kunde vil jeg kunne hente en monteringsvejledning, så jeg kan installere min brugerdefinerede carport, hvis jeg har betalt.(must have)	Lav html side, hvor kunden kan se sine ordre, og tilhørende controller.  Lav en mapper til at hente information, som skal vises på siden.  Sikre sig at plantegningen kun bliver vist, hvis orden er betalt.  Lav tilhørende CSS.	Givet, at jeg har betalt for en brugerdefineret carport, og jeg står på visningen af min ordre siden, vil jeg gerne kunne hente en plan til byggelsen af carporten, hvis jeg har betalt.	Small(såfremt us7 er lavet ellers medium)
--	---	--	---

## Domæne model og ER diagram:



Vi har en 0,1-1 relation vedrørende requires\_slope og material. Det er valgt således, at hvis der ikke findes en requires\_slope for et materiale, kræver den ikke en hældning.

Vi har valgt, at alle fremmednøgler skal hedde det samme, som det de refererer til.

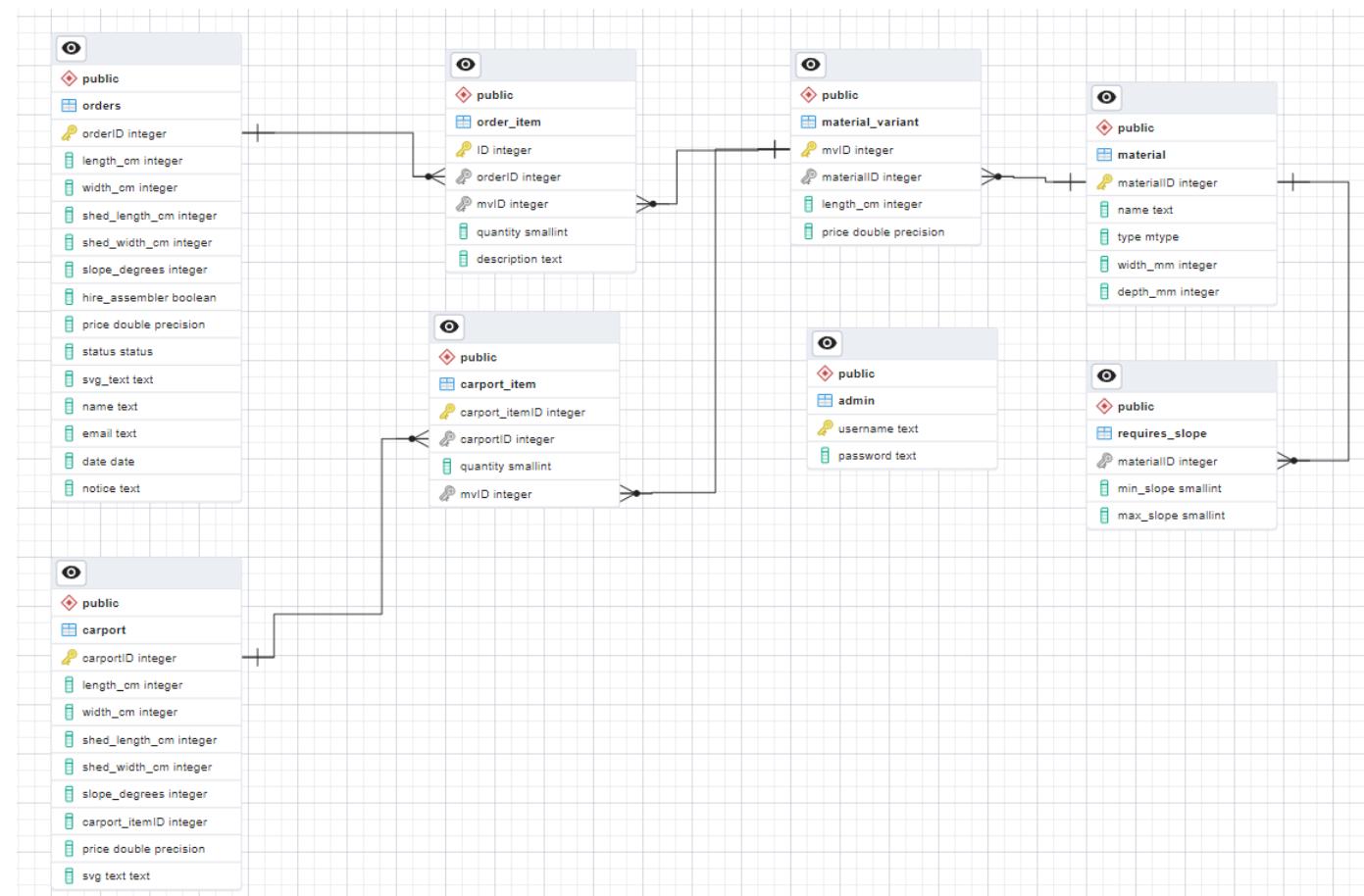
order\_item har 2 fremmednøgler, en til orderID i orders og en mvID til material\_variant's mvID.

material\_variant har en fremmed nøglen materialID til material.

requires\_slope har fremmed nøglen materialID til material.

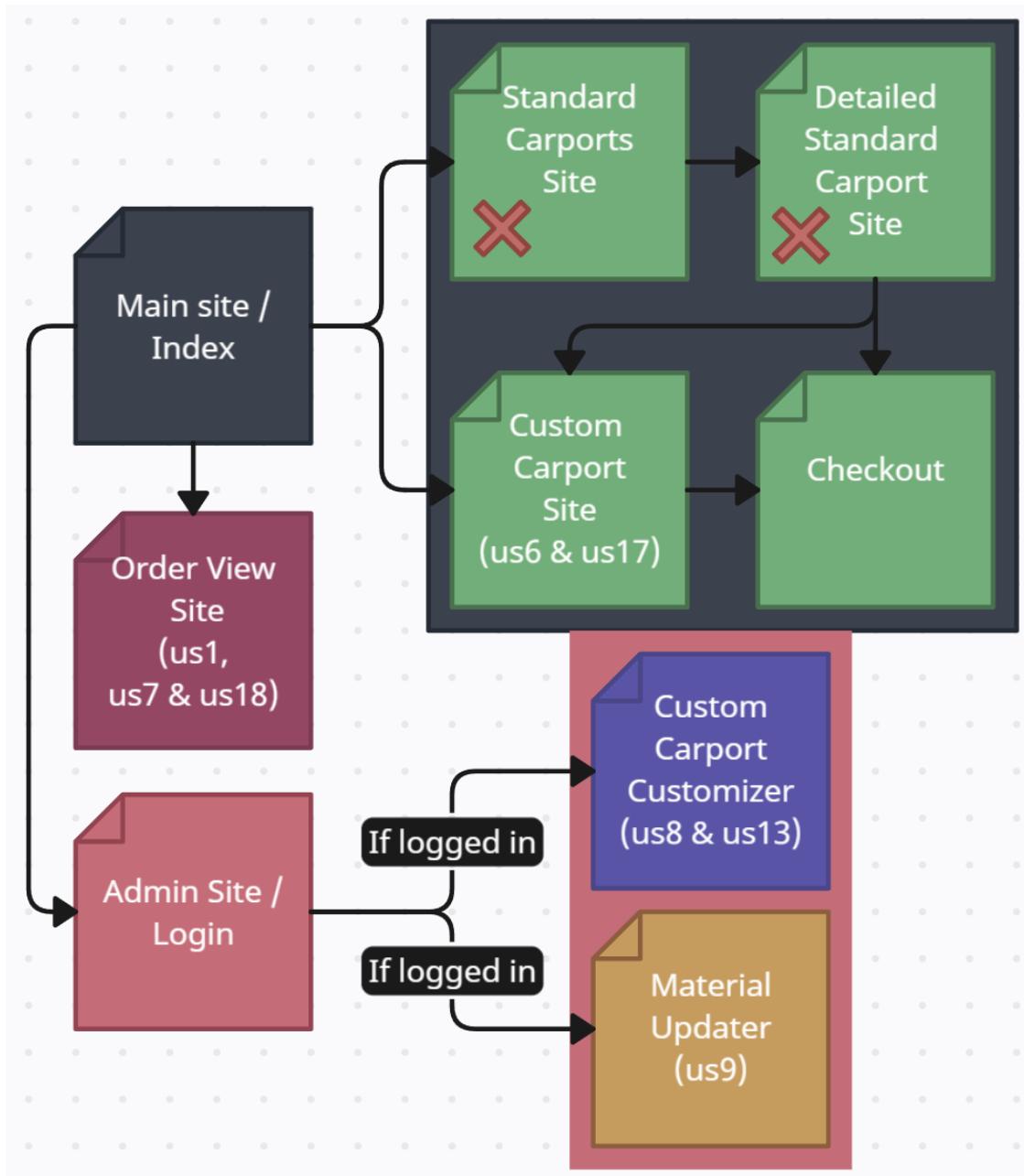
carport\_item har 2 fremmed nøgler, mvID til material\_variant og carportID til carport.

## ER-diagram:



Adminen er en eller eventuelt flere hardcoded users, som sidder i databasen, hvilket den almene kunde ikke har adgang til. Carport og carport\_item var, hvor informationen om vores premade carporte skulle ligge. De bliver ikke brugt og indeholder ingen data.

## Navigationsdiagram:



Vi har valgt at bruge Creately ([Creately | Visual Collaboration & Diagramming Platform](#)) til at designe vores navigationsdiagram, da vi synes, det var et godt alternativ til UML samt hurtigere for os at arbejde i.

Vi har valgt at indikere, hvor vores forskellige user stories bliver løst på siden i diagrammet. De sites, der er markeret med et rødt kryds, er ikke blevet implementeret på rapportskrivningstidspunktet.

Alle vores sider har en fælles navigationbar, som gør det muligt at hoppe mellem main site, admin site og Custom Carport siten.

Som indikeret på diagrammet kan admindelen af siden kun nås, efter man er logget ind med en admin bruger. Der er ikke nogen mulighed for selv at oprette en bruger, men der er på forhånd hardcoded admin logins i databasen.

## Mockups:

Når man designer et produkt, som andre skal gøre brug af, er det meget relevant at starte med en form for prototype af det visuelle udtryk. Dette kan gøres på samtlige måder, men vi har valgt at lave et mockup af nogle af siderne ved hjælp af softwaren Figma.

Disse mockups er til hjælp for os, så vi ved, hvordan vi skal designe hjemmesiden, når vi når til CSS. Samtidig er det et brugbart værktøj til at kommunikere med kunden og diskutere, hvorvidt vi deler vision om, hvad hjemmesiden skal kunne, og hvordan den skal se ud.

Vores mockup skal repræsentere, hvordan vi i sidste ende gerne ville ende med, at produktet skal se ud i et ideelt scenarie. Det er designet ud fra "Fogs" egen hjemmeside, simplificeret.

Mockuppet er designet med brugervenlighed i tankerne og er blevet udviklet ud fra nogle af website design principper såsom gestaltlovene. Herfra, har vi prøvet at skabe et simpelt design med fokus på, at alt, der er relevant for hinanden, er samlet i en boks, samt at interaktive elementer fremgår tydeligt på siden. Vi ønsker samtidig at give hjemmesiden et professionelt udtryk uden at overvælde kunden med for store krav som bruger. Så nogle af brugerens valg er blevet fordelt hen over flere sider, i stedet for at have det hele i en samlet form. Det fulde mockup kan findes vedlagt i projektet ved navn "Mockup".

The image displays two wireframe mockups of a website interface for 'Fog'. Both mockups feature a header with a search bar and navigation links, and a footer with standard delivery options: '30 dages return', 'Klik og hent inden for 3 timer\*', '1-3 dages levering\*', and 'Lån en trailer gratis'.

**Top Mockup:**

- Fog** logo in a blue box.
- A section titled 'Vælg en bestillingstype:' with two options: "'Byg selv' Carport' and 'Udforsk Sortiment'.
- Both options show a small icon of a carport.

**Bottom Mockup:**

- Fog** logo in a blue box.
- A section titled 'Bestil Quick-byg tilbud'.
- Text: 'Med et specialudviklet computerprogram kan vi lynhurtigt beregne prisen og udskrive en skitsetegning på en carport indenfor vores standardprogram. Tilbud og skitsetegning fremsendes med post hurtigt muligt. Ved bestilling medfølger standartbyggevejledning.' and 'Udfyld formularen omhyggeligt og klik på "Bestil tilbud". Felter markeret med \* SKAL udfyldes!'
- Form fields for 'Carport bredde\*', 'Carport længde\*', 'Tagtype\*', 'Taghældning', 'Redskabsrum', 'Redskabsrum bredde\*', 'Redskabsrum længde\*', and 'Evt. bemærkning / Særlige ønsker'.
- A 'Næste' button at the bottom.

## Valg af arkitektur:

Vi har valgt at bruge en form for MVC arkitektur.

Model: Vi bruger vores package "persistence" som indeholder vores mappers, som er de klasser vi bruger som grænseflade mellem vores postgresql database og java.

View: Vi bruger html og css liggende uden for Java strukturen i resources/templates.  
Vores html bliver enhanced ved brug af thymeleaf, som er en kodestruktur, man kan bruge til at lave dynamisk indhold i html, hvis man køre hjemmesiden i Java.

Controller: Vi bruger vores package "controllers", som indeholder vores controllers. Dette er de klasser, som indeholder vores logik fra Java til at vise vores hjemmeside til brugere og inkluderer kunder og sælgere.

Vi har 6 packages med hjælpeklasser, da hjælpeklasser kan reducere gentagende kode meget og eventuelt blive genbrugt senere.

Den første i blandt dem har vi vores "config" package der bliver brugt til, at have indstillinger til vores hjemmeside.

Vores "calculator" package er til for, at kunne give et hurtigt gætteværk på, hvad materiale en bestilling skal bruge og også for at udregne prisen på den.

Vores "svg" package er til for at kunne instantisere en tegning i svg og ved brug af hjælpefunktioner, disse klasser implementere serializable, hvilket gør, at man kan gemme objektet som data.

Den pakke bliver brugt af vores "engine" som er den, der styrer, hvad der bliver tegnet, når man giver den en række data.

Derudover har vi "serialize" som bruges til, at serialisere vores SVG, for at kunne gemme den i databasen.

Til sidst har vi "validators", som bruges til at validere inputs fra brugeren.

Vi har brugt en del klasser til brug som Data Transfer Objects, som vi har lagt i vores package "entities", og i blandt det har vi 2 Enums "Status" og "Mtype", som også eksisterer i vores database.

# Særlige forhold samt Udvalgte kodeeksempler:

De mest væsentlige informationer, der gemmes i vores program er forskellige session attributter, som bruges på de forskellige sites til at gemme information og/eller diktere hvilken information der er tilgængelig for bruger. Herunder beskriver vi kort, hvordan vi bruger disse session attributter i vores program på de forskellige sider.

## Custom Carport Site:

Her bliver en ordreDTO opbygget på baggrund af brugerinputs, som vi gemmer mellem staderne som en session attribute: `current_order`.

Efter kunden har indtastet alt nødvendig information, bliver der vist en opsummering ved hjælp af `current_order`:

```
<table>
  <tr>
    <th>Carport Detail:</th>
  </tr>
  <tr><td>Længde: <span th:text="${session.current_order.getLengthCm()}"></span></td></tr>
  <tr><td>Bredde: <span th:text="${session.current_order.getWidthCm()}"></span></td></tr>
  <tr><td>Skur længde: <span th:text="${session.current_order.getShedLengthCm()}"></span></td></tr>
  <tr><td>Skur bredde: <span th:text="${session.current_order.getShedWidthCm()}"></span></td></tr>
  <tr><td>Tag hældning i grader (hvis valgt ellers automatisk 0): <span th:text="${session.current_order.getSlopeDegrees()}"></span> </td></tr>
</table>
<table>
  <tr><th>User:</th>
  </tr>
  <tr><td>Navn: <span th:text="${session.current_order.getName()}"></span></td></tr>
  <tr><td>Email: <span th:text="${session.current_order.getEmail()}"></span></td>
  </tr>
</table>
```

Efter kunden har double checked sin information og evt. skrevet en kommentar, bliver ordreDTO'en skrevet ned i databasen.

## Custom Carport Customizer:

Det interessante at snakke om her er at, vi har 4 vigtige forskellige session attributter:

`chosen_order`, `costumer_orders`, `bill_of_materials` og `material_list`.

Hvor `chosen_order` er den første som tages i brug, og får siden til at gå mellem forskellige "stadier" alt efter om den er `null` eller ej.

Første stadie er når `chosen_order` er `null`. I første stadie vil der på siden blive spurgt om input fra bruger, hvor der kan indtastes et order id direkte eller der kan søges efter enten name eller email på den ordre man vil arbejde på som kan ses i koden nedenfor:

```
<div class="no_order_chosen" th:if="${session.chosen_order == null}">
    <div>
        <p> Enter ID for the order you want to edit. Alternately enter name or email to see if there is any orders associated with them</p>
    </div>
    <p th:if="${message != null}" th:text="${message}"></p>
    <form th:action="@{/submitOrderID}" method="post">
        <label for="orderID_input">Enter Order ID:</label>
        <input type="number" id="orderID_input" name="orderID_input"/>
        <button type="submit">Submit</button>
    </form>
    <div class="costumer_order" th:if="${session.costumer_orders == null || session.costumer_orders.isEmpty() == true}">
        <h1> Input Costumer info: </h1>
        <form th:action="@{/submitCostumerName}" method="post">
            <label for="name_input">Enter Costumer Name:</label>
            <input type="text" id="name_input" name="name_input"/>
            <button type="submit">Submit</button>
        </form>
        <br><br/>
        <form th:action="@{/submitCostumerEmail}" method="post">
            <label for="email_input">Enter Costumer Email:</label>
            <input type="email" id="email_input" name="email_input" value="email"/>
            <button type="submit">Submit</button>
        </form>
    </div>
</div>
```

Hvis man søger på navn eller email vil resultaterne ende i `costumer_orders` og blive vist på siden. Hvor efter der er muligt at vælge hvilken ordre man vil arbejde på, og `chosen_order` bliver sat til den ordre hvis ID bliver indstillet(submitted).

I tilfælde af at der ikke bliver fundet nogen ordre som passer på det navn / email som der blev søgt på, vil der blive sendt en besked ud til bruger om dette:

```
<p th:if="${message != null}" th:text="${message}"></p>
```

Hvorefter man kan søge igen.

Andet stadi er når `chosen_order` ikke er `null` (altså sat med en ordre). Der vil komme en form frem, der automatisk udfyldes med informationen fra den valgte ordre og kan nu redigeres efter behov. Nedenstående kode exemple viser en bid af koden der gør det muligt at redigere orden:

```
<h1>Here we can customize the costumers order!</h1>
<p th:if="${message != null}" th:text="${message}"></p>
<form th:action="@{/updateOrder}" method="post">
    <h5>Edit the order length in cm:</h5>
    <input type="number" id="newOrderLength" name="new_length_input" th:value="${session.chosen_order.getLengthCm()}" />
    <h5>Edit the order width in cm:</h5>
    <input type="number" id="newOrderWidth" name="new_width_input" th:value="${session.chosen_order.getWidthCm()}" />
    <h5>Edit the order shed length in cm:</h5>
    <input type="number" id="newOrderShedLength" name="new_shed_length_input" th:value="${session.chosen_order.getShedLengthCm()}" />
    <h5>Edit the order shed width in cm:</h5>
    <input type="number" id="newOrderShedWidth" name="new_shed_width_input" th:value="${session.chosen_order.getShedWidthCm()}" />
    <h5>Edit the order slope degrees:</h5>
    <input type="number" id="newSlopeDegrees" name="new_slopeDegrees_input" th:value="${session.chosen_order.getSlopeDegrees()}" />
    <h5>Edit if the order need an assembler:</h5>
    <label>
        <input type="checkbox" id="newHasAssembler" name="new_has_assembler_input" th:checked="${session.chosen_order.isHasAssembler()}" />
    </label>
    <h5>Edit the Order price:</h5>
    <p th:if="${price_message != null}" th:text="${price_message}"></p>
    <input type="number" id="newPrice" name="new_price_input" th:value="${session.chosen_order.getPrice()}" />
    <h5>Edit the Order status:</h5>
    <select th:if="#{strings.equals(session.chosen_order.getStatus().toString(), 'initialised')}" name="edited_order_status">
        <option name="initialised" th:text="initialised">initialised</option>
        <option name="accepted" th:text="accepted">accepted</option>
        <option name="paid" th:text="paid">paid</option>
        <option name="processing" th:text="processing">processing</option>
        <option name="waiting_for_customer" th:text="'waiting_for_customer'">waiting_for_customer</option>
    </select>
```

Derudover vil der også være en save ordre knap og en knap (generate bill of material) der leder til Bill of Material edit hvor man kan generere en stykliste og en plantegning baseret på de nuværende størrelsesforhold på carporten som er gemt på ordren.

### Order View Site:

Her har kunden mulighed for at følge med på hvilket stadi deres ordre er på. Til det bruger vi session attributten: `customer_order`. Der to måder at finde sin ordre på:

Bruger kan indtaste sin ordre ID eller email. Hvis ordre id bruges, vil den forsøge at finde en ordre i databasen der matcher den id og der efter bliver `customer_order` sat til den fundne ordre. Hvis e mail bruges vil den finde de ordre der matcher den indtastede information, hvis der kun bliver fundet en vil den blive sat til `customer_order` i tilfælde af der blev fundet mere end 1 der matchede informationen, vil en liste blive vist hvor man kan vælge den man er interesseret i, `customer_order` vil blive sat til den valgte ordre.

Det er også muligt at komme ind på ordre View siden med et link, exemple herunder:



Efter at `customer_order` er blevet sat vil der komme general information of ordren frem, og hvis ordrens status er sat til paid. Vil der også være plantegning og stykliste til rådighed, hvis de er blevet lavet endnu.

### *Bill of Material edit:*

Det er her hvor `bill_of_materials` og `material_list` bliver taget i brug. Når der trykkes på generate bill of material knappen, vil der automatisk blive tjekke om der allerede er en `bill_of_materials` og en svg string gemt på ordren, hvis de bliver fundet vil de automatisk blive vist og gøre det muligt at ændre på den efter behov.  
Hvis der ikke allerede er en bill of material og svg string gemt på ordren. Vil der automatisk blevet tjekket om der skulle ligge en i databasen. I tilfælde af at der heller ikke er en i databasen vil der blive generet en midlertidig bill of material og vi gemmer den med det samme i databasen:

```
// if there is no bill of material list, we try and get one.
if (billOfMaterials == null) {
    try {
        billOfMaterialListFromDB = OrderItemMapper.getOrderItemsByOrderID(currentOrder.getId(), connectionPool);
        billOfMaterials = billOfMaterialListFromDB;
    } catch (DatabaseException e) {
        ctx.attribute("message", "was unable to find order items" + e.getMessage());
        ctx.render( filePath: "billOfMaterialEditSite.html");
        return;
    }
}
// if there is no bill of material list and nothing was found in the db, we generate one and save it.
if ((billOfMaterials == null || billOfMaterials.isEmpty()) && (billOfMaterialListFromDB == null || billOfMaterialListFromDB.isEmpty())) {
    try {
        billOfMaterials = Calculator.generateBillOfMaterials(currentOrder, connectionPool);
        MaterialsMapper.deleteOrderItemsByOrderID(currentOrder.getId(), connectionPool);
        OrderItemMapper.saveBillOfMaterials(billOfMaterials, currentOrder.getId(), connectionPool);
    } catch (DatabaseException e) {
        ctx.attribute("message", "unable to get any information" + e.getMessage());
        ctx.render( filePath: "billOfMaterialEditSite.html");
        return;
    }
}
```

Hvorefter den vil blive vist på siden, sammen med en plantegning af carporten hvis der er en svg tekst til rådighed og hvis en liste af materialer der er til rådighed (bliver holdt i session attributen: `material_list`). Materialerne kan man tilføje efter behov til styklisten, hvis man forsøger at tilføje et materiale som allerede er i styklisten vil den i stedet tælle materialet 1 op og blive en besked til bruger om at materialet allerede er i styklisten:

```
private static void addMaterialToBillOfMaterial(Context ctx) {
    int index = Integer.parseInt(ctx.formParam( key: "selected_material_input"));
    List<MaterialDTO> billOfMaterials = ctx.sessionAttribute( key: "bill_of_materials");
    List<MaterialDTO> listOfMaterials = ctx.sessionAttribute( key: "material_list");
    MaterialDTO chosenMaterial = listOfMaterials.get(index);
    int checkVar = 0;
    for (int i = 0; i < billOfMaterials.size(); i++) {
        if (billOfMaterials.get(i).getMaterialVariantID() == chosenMaterial.getMaterialVariantID()) {
            checkVar++;
            int originalAmount = billOfMaterials.get(i).getAmount();
            billOfMaterials.get(i).setAmount(originalAmount + 1);
            ctx.attribute("message", "The chosen material was already in the bill of material, 1 was added to the quantity");
        }
    }
    if (checkVar < 1) {
        billOfMaterials.add(chosenMaterial);
    }
    ctx.sessionAttribute("bill_of_materials", billOfMaterials);
    ctx.render( filePath: "billOfMaterialEditSite.html");
}
```

## Engine:

Jeg vil gerne tage fat i `drawCarportDraft1` hvor spærene bliver udregnet.

Den første for loop bruges til at loope vertikalt, så vi tager en række af gangen.

Derefter sætter den variablen currentBeam, som bruges til, at holde styr på længden af spæret.

Den går så fra pæl til pæl og ser om den er i stand til, at kunne nå med et længere spær.

Derefter er der en speciel regel som gør spærret kortere, hvis det sidste spær bliver for kort til at ligge på 2 stolper.

Til sidst har vi speciel reglen som ser om det er det sidste spær hvorefter den tegner firkanten og lægger bjælken ind i styklisten.

til sidst bliver currentBeam resettet og startpunktet forskudt.

```
//// Beams
for(float vertical = sidePillarDistance; vertical < height; vertical += verticalPillarDistance){
    float currentBeam = frontPillarDistance;
    for(float horizontal = 0; horizontal < width ;){
        while(anyBigger(beams, size: currentBeam + horizontalPillarDistance)){
            currentBeam += horizontalPillarDistance;
        }
        if(width - horizontal > 2*horizontalPillarDistance && width - horizontal <= 3*horizontalPillarDistance){
            currentBeam = horizontalPillarDistance;
        }
        if(currentBeam > width - horizontal){
            currentBeam = width - horizontal;
        }
        svg.drawRect(horizontal, x2: horizontal + currentBeam, y: vertical - beamWidth / 2, y2: vertical + beamWidth / 2);
        addItem(items, getBestMaterialLengthOver(beams, currentBeam), amount: 1);
        horizontal += currentBeam;
        currentBeam = 0;
    }
}
```

## Hvordan vi har valgt at håndtere fejl / Exceptions:

På alle vores sider håndtere vi exceptions ved catche dem på controller og i nogle tilfælde mapper niveau, der håndterer vi dem ved at sende en fejlmeldelse ud til bruger:

```
<p th:if="${message != null}" th:text="${message}"></p>
```

Vi har lavet vores egen exception: "Database Exception" som vi ,udover standart exceptions, kaster fra vores forskellige mappere. Ovenstående koden er et eksempel på hvordan vi viser informationen på vores sider, den vil komme med en bestemt besked alt efter hvor i koden den kommer fra.

```
} catch (DatabaseException e) {
    ctx.attribute("message", "No Orders matched what you entered");
    backToOrderSite(ctx);
}
```

Ovenstående er et eksempel på hvordan vi "catcher" vores exception, sender en besked afsted/med og router et sted hen.

## Hvordan vi har kørt validering på brugerinput:

Fordi, at vi håndterer meget brugerinput i fra både kunder men også admin, så er det relevant at vi kører noget validering på input. Dette har vi gjort på forskellige måder. Primært, at vi gør brug af en "Validator" klasse, under vores "Validators" package. I den klasse, har vi forskellige metoder til forskellige input, og hvor de kommer fra.

### **Validering af brugerinput hvor vi kunne have et identisk resultat eller et andet skal udskiftes fremfor brugerens input, hvis de indsætter noget der er ugyldigt:**

Der er nogle steder på vores hjemmeside hvor, at brugeren kan slette et input eller givet et ugyldigt et. F.eks. i vores AdminController hvor man kan redigere et materiale. Der vil vi gerne undgå at man bare ødelægger al dataen ved enten at slette informationen ved at beholde det blank, eller man indsætter ugyldig data. Her bruger vi de forskellige slags metoder ved samme navn "userInput" og overholder den med forskellige parametre.

Metoderne gør brug af en anden valideringsmetode, der kan ses længere nede, og tager to input. Det ene er brugerens input, og det andet er, hvad der skal indskrives i tilfældet af at brugers input ikke kan bruges.

```
2 usages  ± Nicklas W
public static String userInput(String formParam, String alt){
    if(validateString(formParam)){
        return formParam;
    }
    return alt;
}

11 usages  ± Nicklas W
public static int userInput(String formParam, int alt){
    if(validateInt(formParam)){
        return Integer.parseInt(formParam);
    }
    return alt;
}

3 usages  ± Nicklas W
public static double userInput(String formParam, double alt){
    if(validateDouble(formParam)){
        return Double.parseDouble(formParam);
    }
    return alt;
}
```

## Regulær validering af brugerinput:

I de andre tilfælde hvor, at vi bare gerne vil validere et normalt brugerinput i form af en string, har vi også skrevet nogle metoder:

```
1 usage  ± Nicklas W
public static boolean validateDouble(String str){
    if(str == null || str.isEmpty()){
        return false;
    }

    if(!containsOnlyNumbersAndDot(str)){
        return false;
    }

    return true;
}

1 usage  ± Nicklas W
public static boolean validateInt(String str){
    if(str == null || str.isEmpty()){
        return false;
    }

    if(containsOnlyNumbers(str)){
        int number = Integer.parseInt(str);
        if(number <= 0){
            return false;
        }
        return true;
    }

    return false;
}

3 usages  ± Nicklas W
public static boolean validateString(String str){
    if(str == null || str.isEmpty()){
        return false;
    }
    return true;
}
```

Hertil har vi også lavet nogle hjælpemetoder. Den første “containsOnlyNumbersAndDot” gør brug af en regular expression, hvor vi tjekker om strengen kun indeholder numre og maks et punktum. I “containsOnlyNumbers” gør vi brug af Character klassens “isDigit” metode, til at tjekke om hvert tegn i strengen er et tal.

```
1 usage new *
public static boolean containsOnlyNumbersAndDot(String str){
    return str.matches( regex: "[0-9]+(\.[0-9]+)?");
}

1 usage new *
public static boolean containsOnlyNumbers(String str){
    for(char c : str.toCharArray()){
        if(!Character.isDigit(c)){
            return false;
        }
    }
    return true;
}
```

#### Validering direkte i HTML:

Der er også blevet gjort brug af en form for validering ved hjælp af Html. Html har dens egne attributter som kan sættes på et tag, og her bruger vi to som hjælper med at sørge for, at vi får den ønskede data. Der bruges “type” til at definere typen af input vi ønsker, og der gøres brug af “required” så det er umuligt at efterlade feltet tomt.

```
<form method="post" id="costumerDetail" action="/checkout">
<div name="contact_form_thing" class="center">
    <p> Kontaktoplysninger:</p>
    <div name="nameBox">
        <input type="text" placeholder="navn" name="costumer_name" required>
    </div>
    <div>
        <input type="email" placeholder="email" name="user_email" required>
        <div th:if="emailError != null">
            <p th:text="${emailError}"></p>
        </div>
    </div>
```

#### Hvordan vi har valgt at lave sikkerhed i forbindelse med login:

Da kunden ikke har en bruger har den ikke sikkerhed, men admin kræver et brugernavn og password for at kunne komme ind og få tilgængelighed til crud funktionerne.

Hvilke brugertyper (roller) vi har valgt i databasen, og hvordan de er brugt i jdbc:

Der er to roller i databasen admin og anonym user. admin kan lave alle crud funktionerne. anonym user kan kun læse sine ordre fra databasen hvis nogle er tilgængelige ellers har de ingen adgang til databasen. User medlem af admin, uden admin rettigheder. Det er den rolle som dropletten bruger til at forbinde til databasen, via connectionpoolen i koden. Dette er konfigureret i en servicefil på dropletten.

## Status på implementering:

Vi har nået at lave alle websider, som er på navigations diagrammet, pånær standard carport delen. Denne blev pga flere faktorer nedprioriteret til fordel for custom carport delen. Det kommer også til udtryk i tabellen, med status på vores user stories, nedenfor.

Vi valgte at bruge vores fokus på at få custom carport web siderne op at køre først, da standard carport websiderne ikke var en del af kravene til den oprindelige opgave og derfor vurderet som "nice to have".

Til gengæld er custom carport delen blevet implementeret nogenlunde som planlagt. Vores gruppe har været meget ramt af sygdom, hvilket har forsinket vores proces en del. Dette har resulteret i at nogle dele af vores program ikke blevet lavet, som vi oprindelig gerne ville have det.

Vi ville gerne have sat et mailsystem op, som skulle bruges 2 steder i programmet. Mailsystemet skulle bruges til at sende en mail ud til kunden efter at deres ordre var blevet lavet og godkendt med information derom. Derudover skulle der sendes en mail med generel information om orden og processen samt indeholde et link til orderview hjemmesiden, hvor kunden kan holde øje med status på deres ordrer og senere se deres plantegning og stykliste, efter at deres ordrer er betalt. Det andet sted vi ville have brugt mailsystemet, var til at sælger kunne sende en mail ud til kunden med status på deres ordre, om der er kommet nogle ændringer som fx, hvor i processen deres ordre er eller at deres stykliste er klar og afventer betaling.

I stedet for mail-systemet har vi sat et søgesystem op, som kan bruges til at finde sin ordrer. Man kan finde sin mail ved hjælp af sin order ID, sit navn eller sin email. Det har desværre ikke den samme sikkerhed som mailsystemet, da du nu kan få adgang til andres ordrer. Modsat mail systemet, hvor det gerne kun skulle være dig, som har adgang til dit link til orden. Med det sagt er det en funktionel workaround, der med mere tid vil blive erstattet med mailsystemet.

Det var også planen at styklisten, plantegningen og generel/monterings information skulle skrives ned i pdf format og sendes til kunden, når orden er betalt og plantegningen og

stykklisten er genereret. Dette har vi desværre ikke fået implementeret på afleveringstidspunktet.

Vi gør også opmærksom på, at det på givne tidspunkt ikke er muligt at bestille en carport med skur.

De CRUD funktioner, som vi har brug for, er blevet lavet.

Vi har lavet et mockup til, hvordan vi gerne ville have websiderne til at se ud. Det design har vi forsøgt at holde os til så meget som muligt. Vi har dog ikke haft mulighed for ordentligt at implementere css på alle vores sider.

Vi har nået at fået lavet css til alle sider for at give bruger en god oplevelse og sat ordentlig css op på admin: Material Updater (materialEdit) siden for at vise hvad vi kan.

Nedenfor har vi en tabel over status på vores forskellige user stories med tilhørende acceptance:

User Story:	Acceptance	Status:
<b>1.</b> Som kunde vil jeg gerne se mine ordrer, så jeg kan være sikker på, at jeg har bestilt. (must have)	Givet, at jeg er en kunde, og jeg har bestilt en ordre, så vil jeg gå ind på hjemmesiden og trykke på "view your orders" og så indsætte min order id eller email, og trykke "submit" så jeg kan se statussen på min ordre.	Fuldt implementeret. Kunden kan få adgang til en hjemmeside, som viser kundens information om ordren. De kan få adgang til dem både ved hjælp af et link og ved at taste deres ordre id / email ind.
<b>2.</b> Som kunde vil jeg gerne kunne søge på varer for at finde det rigtige produkt.(nice to have)	Givet at jeg er en kunde, og jeg står på Fogs hjemmeside, kan jeg søge i et søgefelt efter et produkt og ved at trykke 'Enter' på mit tastatur, så vil jeg modtage en liste af søgeresultater der passer til min søgning.	ikke implementeret. Blev nedprioriteret for at vi kunne fokusere på "must have" user stories.
<b>3.</b> Som kunde vil jeg gerne kunne se en liste over alle carporte, for at finde den, som passer mig, så jeg kan bestille den.(nice to have)	Givet at jeg har trykket på en knap på hjemmesiden kan jeg se en liste af alle carports der bliver tilbudt af Fog.	ikke implementeret. Blev nedprioriteret for at vi kunne fokusere på "must have" user stories.

<b>4.</b> Som kunde vil jeg gerne kunne se ændringer på mine ordrer, så jeg kan følge med i hvad salgs personen har aftalt med mig (et mer projekt) (must have)	Givet, at jeg er kunde, og jeg har lavet en bestilling, vil jeg kunne trykke på et link, jeg har modtaget via mail, for at se hvilke ændringer der skulle være sket med min bestilling.	ikke implementeret. Blev nedprioriteret for at vi kunne fokusere på “must have” user stories.
<b>5.</b> Som kunde vil jeg gerne kunne se ændringer på priser over tid for at sikre mig at der ikke er snydt med dem i forhold til tilbud hvis de havde sat prisen op inden.(nice to have)	Givet at jeg er kunde når jeg har trykket på en carport vil jeg gerne kunne se hvor lang tid siden, at prisen på et produkt er blevet ændret.	ikke implementeret. Blev nedprioriteret for at vi kunne fokusere på “must have” user stories.
<b>6.</b> Som kunde vil jeg gerne kunne bestille en carport efter mine egne mål, for at finde den, som passer mig. (must have)	Givet, at jeg er kunde, og jeg sidder på “Tilpas carport”-siden, vil jeg kunne tilpasse egne mål og materialer, og derefter lave en bestilling, ved at trykke på “Bestil”-knappen, som derefter sender min bestilling videre til Fog.	Fuldt implementeret: Kunden kan nu gå ind på hovedhjemmesiden, og starte med at tilpasse deres carport, vælge mellem mål og bestille den.
<b>7.</b> Som kunde vil jeg gerne kunne se en liste over materialer som carporten består af, efter jeg har betalt. (must have efter us8)	Givet, at jeg er kunde, og jeg har lavet en bestilling som jeg har betalt for, og betalingen er gået igennem hos Fog. Så vil jeg kunne trykke på et link jeg har modtaget via mail, som sender mig ind på en hjemmeside med en kvittering og en liste over materialer som min bestilling består af.	fuldt implementeret. Som kunde kan jeg nu få adgang til min stykliste og anden information. Enten ved hjælp af et link eller søge efter deres ordre med et ordre nummer eller en email. Styklisten bliver kun vist, hvis ordrens status er “paid”.
<b>8.</b> Som sælger vil jeg gerne kunne hente en plantegning over carporten, så jeg kan få en idé om størrelsesforholdet. (must have)	N/A - Manual Testing required	fuldt implementeret sælger kan generere plantegningen ved at trykke på en knap og vise den på hjemmesiden og gemme svg teksten, som bliver brugt til at vise billedet, til ordren.

<p><b>9.</b> Som sælger vil jeg gerne kunne redigere/tilføje/fjerne produkter så jeg kan holde hjemmesiden opdateret.(nice to have)</p>	<p>Givet at jeg er admin når jeg vælger et produkt, vil jeg gerne have muligheden for at kunne redigere det fra pris til beskrivelse. Givet at jeg er admin, vil jeg gerne kunne slette det valgte produkt eller tilføje et nyt et.</p>	<p>fuldt implementeret. Sælger har mulighed for at adde nye materialer til databasen. Sælger kan også redigere og fjerne materialer.</p>
<p><b>10.</b> Som sælger vil jeg gerne kunne blokere IP adresser, så jeg kan forhindre DDOS angreb. (nice to have)</p>	<p>Givet, at jeg er sælger, og sidder på “administrations” siden, så vil jeg kunne blokere adgang for en bestemt IP-adresse, ved at vælge den ud fra en liste af trafik, for at mindske muligheden for at databasen bliver spammed.</p>	<p>ikke implementeret. Blev nedprioriteret for at vi kunne fokusere på “must have” user stories.</p>
<p><b>11.</b> Som kunde vil jeg gerne kunne sortere efter pris, for at nemmere kunne vælge økonomisk.</p>	<p>Givet at jeg er kunde når jeg søger på carports vil jeg gerne have muligheden for at kunne sortere efter pris.</p>	<p>ikke implementeret. Blev nedprioriteret for at vi kunne fokusere på “must have” user stories.</p>
<p><b>12.</b> Som kunde vil jeg gerne kunne sortere efter navn, for at nemmere kunne finde et bestemt produkt.</p>	<p>Givet at jeg er kunde når jeg søger på carports vil jeg gerne have muligheden for at kunne sortere efter navn.</p>	<p>ikke implementeret. Blev nedprioriteret for at vi kunne fokusere på “must have” user stories.</p>
<p><b>13.</b> Som sælger, skal jeg kunne se en stykliste med al information om bestillingen, så jeg ved hvilke dele, der skal bruges, med deres varenummer, antal, hjælpetekst, beskrivelse, og mål.(must have)</p>	<p>Givet, at jeg er sælger, og jeg står på ”tilpasning af kundes carport”-siden, og jeg har fået indsat al data fra kundens ønsker, kan jeg trykke ”Generer Stykliste” og modtage en liste med al information om de dele der skal bruges og antal.</p>	<p>fuld implementeret. Sælger kan generere en stykliste. Sælger kan se og ændre på de individuelle materialer i styklisten. Sælger kan se en liste over alle materialer der er til rådighed og adde dem til styklisten hvis der vurderes et behov.</p>

<p><b>14.</b> Som sælger, når jeg bruger hjemmesiden til at generere kundens carport, så ved hjemmesiden om hvorvidt taget er med hældning eller ej, baseret på materialet for taget, så jeg kan sikre mig, at jeg får sat den hældning, kunden ønsker.(nice to have, da vi i første omgang skal have implementeret en med fladt tag)</p>	<p>Givet at jeg er kunde når jeg selv designs min carport vil jeg gerne have muligheden for at kunne vælge materialer til taget som kan have hældning eller ej.</p>	<p>ikke implementeret. Blev nedprioriteret for at vi kunne fokusere på “must have” user stories.</p>
<p><b>15.</b> Som kunde vil jeg kunne få højere pris baseret på min lokalitet fra FOG Lagerhus. (nice to have)</p>	<p>Givet at jeg er kunde og bestiller fra Jylland mens jeg er i Sjælland, vil jeg gerne have højere priser for carporten.</p>	<p>ikke implementeret. Blev nedprioriteret for at vi kunne fokusere på “must have” user stories.</p>
<p><b>16.</b> Som kunde vil jeg gerne kunne se en detaljeret beskrivelse af carporten når jeg har valgt den.(nice to have)</p>	<p>Givet at jeg som kunde har trykket på en carport vil jeg på den carports side kunne se en detaljeret beskrivelse af carporten.</p>	<p>ikke implementeret. Blev nedprioriteret for at vi kunne fokusere på “must have” user stories.</p>
<p><b>17.</b> Som kunde vil jeg gerne kunne vælge om jeg hyrer FOG til at sætte carporten op for mig eller gør det selv.(must have)</p>	<p>Givet at jeg som kunde har valgt eller designet min egen carport vil jeg gerne have muligheden for at kunne bede Fog om at sætte den op for mig eller ej.</p>	<p>fuldt implementeret. Når man bestiller en carport, er der mulighed for at tjekke en checkbox, hvorvidt man ønsker en fra FOG til at sætte carporten op for en eller ej.</p>
<p><b>18.</b> Som kunde vil jeg kunne hente en monteringsvejledning, så jeg kan installere min brugerdefinerede carport, hvis jeg har betalt.(must have)</p>	<p>Givet at jeg har betalt for en brugerdefineret carport, vil jeg gerne kunne hente en plan til byggelsen af carporten, hvis jeg har betalt.</p>	<p>fuldt implementeret. Efter man har betalt for sin carport er en plantegning og stykliste tilgængelig for kunden. Hvilke kan bruges som en vejledning til at sætte sin nye carport op (hvilken man ikke ”behøver” hvis man har valgt at fog sætter ens carport op)..</p>

# Kvalitetssikring (Test):

## Automatiserede tests:

Vi har ikke lavet tests på controllers grundet naturen bag funktionerne, da de har brug for en brugergrænseflade for at kunne fungere.

Hver branch er blevet gennemtjekket af branch ejeren før et merge bliver anmodet, derefter bliver de (andre end ejeren) ud af de 3 fletnings-ansvarlige (Nicklas, Patrick eller Nicolai) sat på til review af fletningen. Når en af dem så manuelt har testet branchen og fjerner fejl i branchen som nødvendigt.

Når det var gjort blev det godkendt og sendt til dev branchen hvor man tester igen om altting virker og til sidst smider det ind i main branchen.

Vi valgte denne metode på grund af naturen på projektet, da automatiske tests ikke kan blive gjort på brugergrænseflader med den teknologi, vi er i besiddelse af.

Package	Beskrivelse
engine	<ul style="list-style-type: none"> <li>- Engine</li> <li>- all_setup(): Gælder alle tests: <ul style="list-style-type: none"> <li>- <i>Tjek om der er forbindelse til en lokal developer database.</i></li> <li>- getBeamID(): Tjekker om den returnerer et tal højere end 0. <ul style="list-style-type: none"> <li>- <i>For at forhindre mængden af andet setup nødvendigt, er tjekket baseret på goodwill, hvilket vil sige man stoler på, at der ikke er blevet gjort genveje.</i></li> </ul> </li> <li>- getPillarID(): Tjekker om den returnerer et tal højere end 0. <ul style="list-style-type: none"> <li>- <i>For at forhindre mængden af andet setup nødvendigt, er tjekket baseret på goodwill, hvilket vil sige man stoler på, at der ikke er blevet gjort genveje.</i></li> </ul> </li> <li>- drawCarportDraft1(): Tjekker, at den ikke kaster en fejl, og den returnere data. <ul style="list-style-type: none"> <li>- <i>For at forhindre mængden af andet setup nødvendigt, er tjekket baseret på goodwill, hvilket vil sige man stoler på, at der ikke er blevet gjort genveje.</i></li> </ul> </li> </ul> </li> </ul>
entities	<ul style="list-style-type: none"> <li>- BeamDTO <ul style="list-style-type: none"> <li>- testEquals(): laver 3 BeamDTO'er og tjekker om de 2 der er identiske ses som ens og den sidste ikke gør.</li> </ul> </li> </ul>
persistence	<ul style="list-style-type: none"> <li>- MaterialsMapper <ul style="list-style-type: none"> <li>- all_setup(): Gælder alle tests: <ul style="list-style-type: none"> <li>- <i>Tjek om der er forbindelse til en lokal developer database.</i></li> </ul> </li> <li>- getMaterialInfoByType(): Tjekker om data bliver trykket og kun den rigtige type. <ul style="list-style-type: none"> <li>- <i>ret robust, kun mere robust hvis andre funktioner var brugt.</i></li> </ul> </li> <li>- getAllMaterialInfo(): Tjekker om data bliver trykket ned. <ul style="list-style-type: none"> <li>- <i>ret robust, kun mere robust hvis andre funktioner var brugt.</i></li> </ul> </li> <li>- getAllMaterials(): Tjekker om data bliver trykket ned. <ul style="list-style-type: none"> <li>- <i>ret robust, kun mere robust hvis andre funktioner var brugt.</i></li> </ul> </li> <li>- updateMaterial(): Tager bredden på vores testPillar og giver den en tilfældig ny bredte, som ikke er den samme som på databasen. herefter tjekkes om funktionen resultere i sandt og om materialet er opdateret på databasen ved hjælp af getMaterialByld funktionen, som testes senere. <ul style="list-style-type: none"> <li>- <i>ret robust, kun mere robust hvis flere tjeneste blev sat..</i></li> </ul> </li> <li>- getMaterialByld(): Tjekker om test materialet kan hentes fra databasen og om det er ens med den vi i forvejen har. <ul style="list-style-type: none"> <li>- <i>ret robust, kun mere robust hvis andre funktioner var brugt.</i></li> </ul> </li> <li>- addMaterial(): Tjekker om addMaterial funktionen returnerer sandt og om den har tilegnet materialet en id. <ul style="list-style-type: none"> <li>- <i>For at forhindre mængden af andet setup nødvendigt, er tjekket baseret på goodwill, hvilket vil sige man stoler på, at der ikke er blevet gjort genveje.</i></li> </ul> </li> <li>- removeMaterial(): Tjekker om removeMaterial funktionen returnerer sandt og om den har sat dens ID til 0. <ul style="list-style-type: none"> <li>- <i>For at forhindre mængden af andet setup nødvendigt, er tjekket baseret på goodwill, hvilket vil sige man stoler på, at der ikke er blevet gjort genveje.</i></li> </ul> </li> <li>- OrderItemMapper - startet på en, men blev efterladt.</li> <li>- OrderMapper <ul style="list-style-type: none"> <li>- addOrder(): Tjekker om en ordre kan blive tilføjet og om den har en id. <ul style="list-style-type: none"> <li>- <i>For at forhindre mængden af andet setup nødvendigt, er tjekket baseret på goodwill, hvilket vil sige man stoler på, at der ikke er blevet gjort genveje.</i></li> </ul> </li> <li>- updateOrder(): Tjekker om vi kan ændre værdier for en ordre og om den returnerer sandt. <ul style="list-style-type: none"> <li>- <i>For at forhindre mængden af andet setup nødvendigt, er tjekket baseret på goodwill, hvilket vil sige man stoler på, at der ikke er blevet gjort genveje.</i></li> </ul> </li> <li>- getOrdersBySearchDTO(): laver en søgning som kun skulle finde vores test ordre og ser om den returnerer en ordre. <ul style="list-style-type: none"> <li>- <i>ret robust, kun mere robust hvis flere tjenester blev sat og er afhængig af addOrder() virker..</i></li> </ul> </li> </ul> </li> </ul> </li></ul>

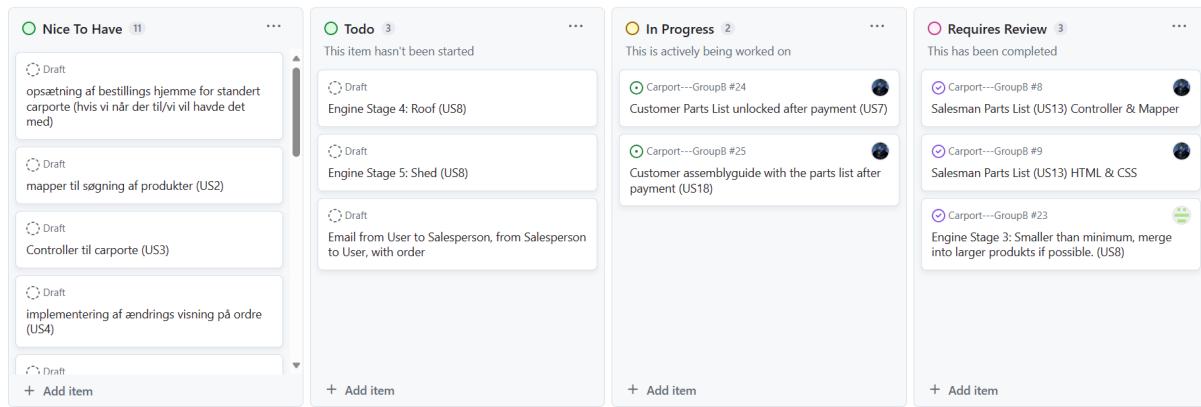
## User Acceptance tests:

User Story	UAT
1. Som kunde vil jeg gerne se mine ordrer, så jeg kan være sikker på, at jeg har bestilt. (must have)	godkendt.
2. Som kunde vil jeg gerne kunne søge på varer for at finde det rigtige produkt.(nice to have)	godkendt.
3. Som kunde vil jeg gerne kunne se en liste over alle carporte, for at finde den, som passer mig, så jeg kan bestille den.(nice to have)	godkendt.
4. Som kunde vil jeg gerne kunne se ændringer på mine ordrer, så jeg kan følge med i hvad salgs personen har aftalt med mig (et mer projekt) (must have)	godkendt.
5. Som kunde vil jeg gerne kunne se ændringer på priser over tid for at sikre mig at der ikke er snydt med dem i forhold til tilbud hvis de havde sat prisen op inden.(nice to have)	godkendt.
6. Som kunde vil jeg gerne kunne bestille en carport efter mine egne mål, for at finde den, som passer mig. (must have)	godkendt.
7. Som kunde vil jeg gerne kunne se en liste over materialer som carporten består af, efter jeg har betalt. (must have efter us8)	godkendt.
8. Som sælger vil jeg gerne kunne hente en plantegning over carporten, så jeg kan få en idé om størrelsesforholdet. (must have)	godkendt.
9. Som sælger vil jeg gerne kunne redigere/tilføje/fjerne produkter så jeg kan holde hjemmesiden opdateret.(nice to have)	godkendt.
10. Som sælger vil jeg gerne kunne blokere IP adresser, så jeg kan forhindre DDOS angreb. (nice to have)	godkendt.

<b>11.</b> Som kunde vil jeg gerne kunne sortere efter pris, for at nemmere kunne vælge økonomisk.	godkendt.
<b>12.</b> Som kunde vil jeg gerne kunne sortere efter navn, for at nemmere kunne finde et bestemt produkt.	godkendt.
<b>13.</b> Som sælger, skal jeg kunne se en stykliste med al information om bestillingen, så jeg ved hvilke dele, der skal bruges, med deres varenummer, antal, hjælpetekst, beskrivelse, og mål.(must have)	godkendt.
<b>14.</b> Som sælger, når jeg bruger hjemmesiden til at generere kundens carport, så ved hjemmesiden om hvorvidt taget er med hældning eller ej, baseret på materialet for taget, så jeg kan sikre mig, at jeg får sat den hældning, kunden ønsker.(nice to have, da vi i første omgang skal have implementeret en med fladt tag)	godkendt.
<b>15.</b> Som kunde vil jeg kunne få højere pris baseret på min lokalitet fra FOG Lagerhus. (nice to have)	godkendt.
<b>16.</b> Som kunde vil jeg gerne kunne se en detaljeret beskrivelse af carporten når jeg har valgt den.(nice to have)	godkendt.
<b>17.</b> Som kunde vil jeg gerne kunne vælge om jeg hyrer FOG til at sætte carporten op for mig eller gør det selv.(must have)	godkendt.
<b>18.</b> Som kunde vil jeg kunne hente en monteringsvejledning, så jeg kan installere min brugerdefinerede carport, hvis jeg har betalt.(must have)	godkendt.

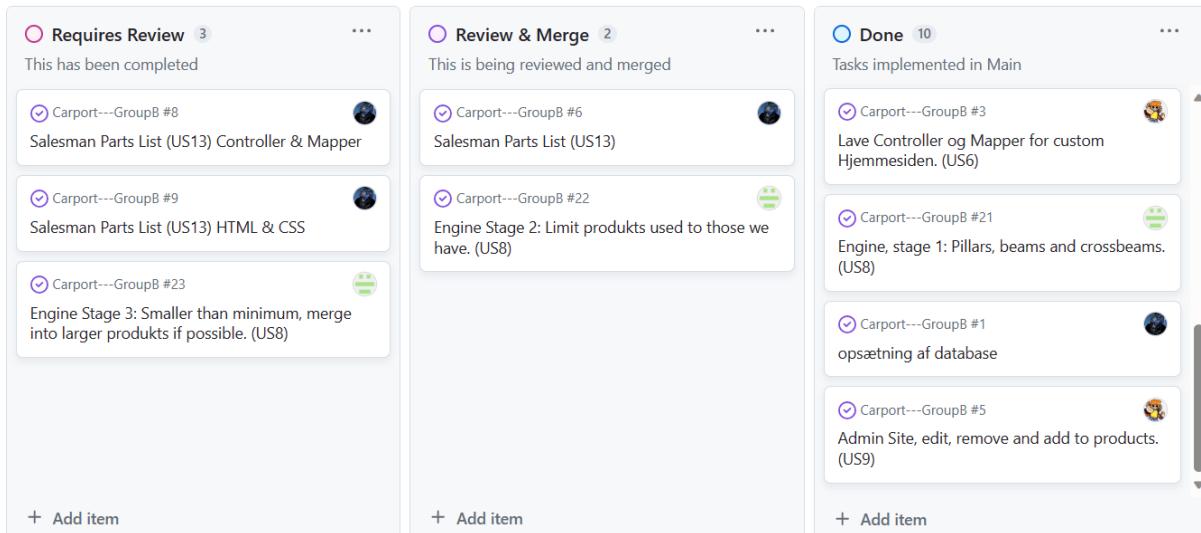
## Proces:

Vores arbejdsproces begyndte med, at vi startede et projekt på GitHub. Her oprettede vi et kanban board og satte alle vores user stories ind. Dem, der var for store, blev nedbrudt i mindre cases. Efter det, vurderede vi, hvor vigtig den enkelte user story var og gav den en must have / nice to have status. Derefter kunne gruppemedlemmerne melde sig på de cases, som de syntes gav mening for dem. I forbindelse med at et gruppemedlem påtog sig en case, oprettede de en branch på projektet, hvis navn reflekterede den case, de havde planer om at arbejde på. Herefter løste de opgaven på branchen og sikrede sig at vores forskellige diagrammer dækkede over den case, som var blevet valgt. Hvis diagrammerne ikke er dækkende, vil de blive udvidet eller der vil blive oprettet nye efter behov.



I takt med at de forskellige user stories blev løst, ville de blive rykket over i Requires Review, hvor en anden fra gruppen end den, som var assigned til user story'en, skulle tjekke den igennem før den kunne merges ind i master(main) branchen.

Når review processen for en user story gik i gang, blev den rykket til Review & Merge af personen, som har påtaget sig review opgaven. Hvis koden blev godkendt, blev den til slut merged ind i Master branch og rykket over i Done colonen:



## Arbejdsprocessen faktuelt:

Vi havde i alt 4 faser hvor der blev arbejdet på enten de samme ting igennem alle faserne eller nogle nye.

1. fase:  
fik opsat alt setuppet til projektet, herfra bla. "snakket" med kunden med detaljer om bestilling af og opsætning af carport processen. derfra database opsætning og begyndelse på kodning.
2. fase:  
delvis færdiggørelse af calculatoren og flere dtoer blev oprettet for at understøtte den og en engine.
3. fase:  
admin siden fuldt færdiglavet, css blev påbegyndt på flere af hjemmesiderne.
4. fase:  
codefreeze på kodning med nye ting. herfra udelukkende fokus på bugfixing og minimal viable product.

Vi havde ingen KanBan mester i vores projekt, men KanBan gjorde det meget nemmere at uddeleger opgaverne iblandt os, hvor vi hver især valgte noget vi gerne ville gøre eller lære af. Til vores første møde ville vi høre om, hvad der var vigtigst for projektet og der fik vi at vide, at vi skulle fokusere på et minimum viable product frem for nice-to-have featurene, som vi havde planlagt. Yderligere information om vores beslutninger baseret på møderne kan findes i vores logbog.

Vores team kommunikation kunne have været bedre, men da vi begyndte at møde fast på skolen skete der store fremskridt frem for at sidde på discord. Vi var relativt gode til at møde på aftalt tid og derfra tog projekt fart. Når vi sad fysisk sammen, var det nemmere at spørge hinanden til råds og hjælpe hinanden til forskel fra når man sad hver for sig og håbede på, at andre fra gruppen var online.

## Arbejdsprocessen reflekteret:

Vi havde ikke en KanBan mester rolle. Det nærmeste vi kom på lederskab var direkte demokrati med alle ups and downs, der medfølger. Dette jorde beslutningsprocessen mere kaotisk, når der var splittede meninger om fremgangsmåden.

Vi har konkluderet at vi fremover skal overveje en projektleder. Yderligere under møderne blev vi fortalt, at vi skulle fokusere mere på rapport frem for selve produktet, der helst skulle være et minimum viable. Vi var ikke alle sammen lige spot on med vores estimeringer af tid grundet sygdom. Vi endte derfor med at nedprioriteret eller droppe nice-to-have featurene. Demoen blev filmet og optaget problemfrit og delingen af filer gik godt på github.

Igennem arbejdsprocessen har vi gjort os mange erfaringer. En af de væsentligste var håndtering af exceptions, som ville have ændret på opsætningen af programmet, hvis vi havde haft den ekstra erfaring ved projektstart.

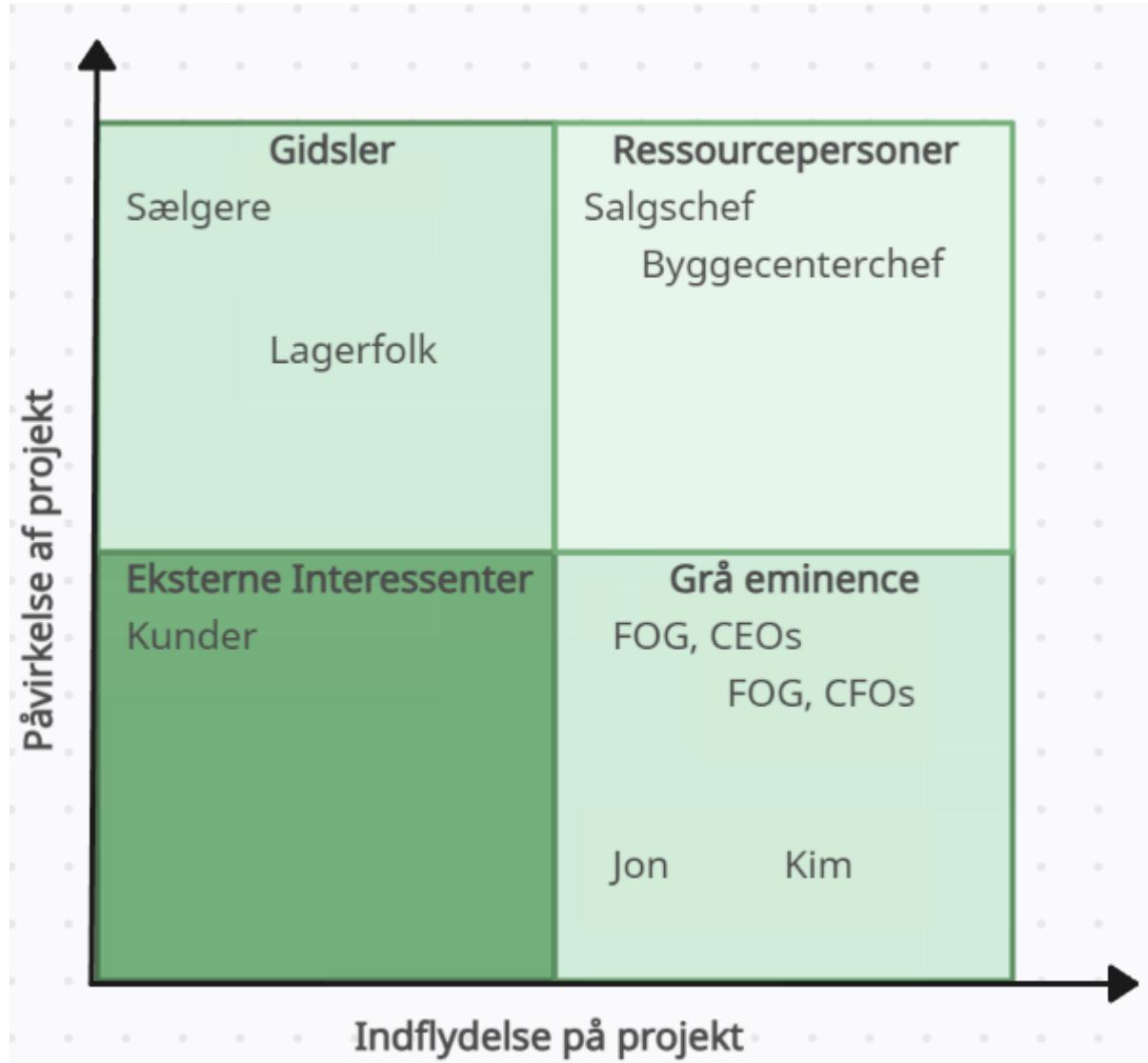
Lige nu har vi kun custom exception(Database Exception) i vores program, som vi kaster fra mappere og catcher i vores controllere. I fremtidige projekter vil vi lave flere sigende custom exceptions og håndtere dem forskelligt alt efter hvilken vi catcher.

Udover det, bruger vi ctx.attribute("message", "besked") til at sende en besked ud til brugere. Den kan dog kun holde en besked. Fremover vil vi lave den om til at tage en liste af Strings, i tilfælde af, at der er mere end én besked, som skal sendes til brugerne og vise dem på siden ved at adde en th:each til div'en/paragrafen. Beskeden sidder i for at kunne vise dem alle beskederne på siden.

Arbejdsfordelingen har også været skæv. Næste gang sørger vi for at fordele arbejdsopgaverne mere ligeligt, så det aldrig er muligt for en person at sige, at de ikke har mere at lave. Havde vi haft den forudsætning fra starten af, var vi muligvis nået længere, end vi gjorde.

## Bilag:

Figur: 1



## Interessentanalyse:

Carport			Udfyldt af: Samlet gruppe	Dato: 21/11/23
Interessent	Interessenten kan opleve følgende fordele ved projektet	Interessenten kan opleve følgende ulemper ved projektet	Samlet vurdering af interessenstens bidrag/position	Håndtering af interessensten

Kunde	Hurtigere betjening. Mindre indtastningsfejl.	Migrations Forstyrrelse.	Lav interesse og indflydelse.	Dem på mail- listen bliver tilsendt en mail med opdatering. Dem på mail- listen kan blive udvalgt til en demo hjemmeside hvor de kan give feedback.
Salgschef	Øget effektivitet for sælgere. Øget kundevenlighed. Højere omsætning i form af flere kunder. Nem opdatering af lager/priser.	Genoplæring af sælgere. Implementering kan tage tid.	Høj interesse og høj indflydelse. Skal medtages i projektet.	Skal informeres og opdateres løbende om projektet. Og involveres i store beslutninger. Dette gøres ved biweekly møde.
Sælgere	Øget effektivitet på arbejdet. Overskuelighed. Færre indtastningsfejl. Mere tid til kundeinteraktion.	Genoplæring og migration forstyrrelser.	Høj påvirkning af projektet, med lav indflydelse. Skal informeres.	Skal informeres om store eller pludselige ændringer. Muligvis med en e-mail eller orienteringsmøde.
CEO	Højere effektivitet for firmaet. Forhøjet aktieværdi.	Periode af tab af penge på produktion.	Lav påvirkning, meget høj indflydelse. Kan stoppe projektet når som helst. Skal høres	Skal konstant høres og deres beslutning er lov. Dette gøres ved at samle manglende beslutninger og muligt holde et møde ugentligt.
CFO	Profit over tid. Forhøjet aktieværdi.	Kost som bliver genvundet af sælger effektivitet.	Lav påvirkning, meget høj indflydelse. Kan stoppe projektet når som helst. Skal høres	Skal konstant høres og deres beslutning er lov. Dette gøres ved at samle manglende beslutninger og muligt holde et møde ugentligt.
Bygge Centerchef	Øget effektivitet for byggecentret. Øget kundevenlighed. Øget produktion. Nem opdatering af lager og nye varer.	Genoplæring af arbejdere. Migrations Periode.	Høj interesse og høj indflydelse. Skal medtages i projektet.	Skal informeres og opdateres løbende om projektet. Og involveres i store beslutninger. Dette gøres ved at tage dem med til det ugentlige møde.

Lagerfolk	Kontinuerligt opdateret data, som passer ind med lagersystemet.	Skal interviewes. Implementering kan tage tid.	Høj påvirkning, men lav indflydelse.	Skal høres om integration med lagersystemet. Dette gøres ved et møde tidligt i processen, et senere i projektet, hvis behov og ugentlig nyhedsbrev, hvor de kan indsende feedback.
-----------	---	--	--------------------------------------	--

for matrix, se [figur 1](#)

# User Stories & Acceptance Criteria

User Story	Acceptance
<b>1.</b> Som kunde vil jeg gerne se mine ordrer, så jeg kan være sikker på, at jeg har bestilt. (must have)	Givet, at jeg er en kunde, og jeg har bestilt en ordre, så vil jeg trykke på et link, jeg har modtaget via mail, som viser mig en hjemmeside med min/mine ordre.
<b>2.</b> Som kunde vil jeg gerne kunne søge på varer for at finde det rigtige produkt.(nice to have)	Givet at jeg er en kunde, og jeg står på Fogs hjemmeside, kan jeg søge i et søgerfelt efter et produkt og ved at trykke "Enter" på mit tastatur, så vil jeg modtage en liste af søgeresultater der passer til min søgning.
<b>3.</b> Som kunde vil jeg gerne kunne se en liste over alle carporte, for at finde den, som passer mig, så jeg kan bestille den.(nice to have)	Givet at jeg har trykket på en knap på hjemmesiden kan jeg se en liste af alle carports der bliver tilbudt af Fog.
<b>4.</b> Som kunde vil jeg gerne kunne se ændringer på mine ordrer, så jeg kan følge med i hvad salgs personen har aftalt med mig (et mer projekt) (must have)	Givet, at jeg er kunde, og jeg har lavet en bestilling, vil jeg kunne trykke på et link, jeg har modtaget via mail, for at se hvilke ændringer der skulle være sket med min bestilling.
<b>5.</b> Som kunde vil jeg gerne kunne se ændringer på priser over tid for at sikre mig at der ikke er snydt med dem i forhold til tilbud hvis de havde sat prisen op inden.(nice to have)	Givet at jeg er kunde når jeg har trykket på en carport vil jeg gerne kunne se hvor lang tid siden, at prisen på et produkt er blevet ændret.
<b>6.</b> Som kunde vil jeg gerne kunne bestille en carport efter mine egne mål, for at finde den, som passer mig, så jeg kan bestille den. (must have)	Givet, at jeg er kunde, og jeg sidder på "Tilpas carport"-siden, vil jeg kunne tilpasse egne mål og materialer, og derefter lave en bestilling, ved at trykke på "Bestil"-knappen, som derefter sender min bestilling videre til Fog.
<b>7.</b> Som kunde vil jeg gerne kunne se en liste over materialer som carporten består af, efter jeg har betalt. (must have efter us8)	Givet, at jeg er kunde, og jeg har lavet en bestilling som jeg har betalt for, og betalingen er gået igennem hos Fog. Så vil jeg kunne trykke på et link jeg har modtaget via mail, som sender mig ind på en hjemmeside med en kvittering og en liste over materialer som min bestilling består af.
<b>8.</b> Som sælger vil jeg gerne kunne hente en plantegning over carporten, så jeg kan få en idé om størrelsesforholdet. (must have)	N/A - Manual Testing required

<b>9.</b> Som sælger vil jeg gerne kunnen redigere/tilføje/fjerne produkter så jeg kan holde hjemmesiden opdateret.(nice to have)	Givet at jeg er logget på som admin og jeg sidder på edit material admin siden, når jeg vælger et produkt, vil jeg gerne have muligheden for at kunne redigere det fra pris til beskrivelse. Givet at jeg er admin, vil jeg gerne kunne slette det valgte produkt eller tilføje et nyt et.
<b>10.</b> Som sælger vil jeg gerne kunne blokere IP adresser, så jeg kan forhindre DDOS angreb. (nice to have)	Givet, at jeg er sælger, og sidder på "administrations" siden, så vil jeg kunne blokere adgang for en bestemt IP-adresse, ved at vælge den ud fra en liste af trafik, for at mindske muligheden for at databasen bliver spammed.
<b>11.</b> Som kunde vil jeg gerne kunne sortere efter pris, for at nemmere kunne vælge økonomisk.	Givet at jeg er kunde når jeg søger på carports vil jeg gerne have muligheden for at kunne sortere efter pris.
<b>12.</b> Som kunde vil jeg gerne kunne sortere efter navn, for at nemmere kunne finde et bestemt produkt.	Givet at jeg er kunde når jeg søger på carports vil jeg gerne have muligheden for at kunne sortere efter navn.
<b>13.</b> Som sælger, skal jeg kunne se en stykliste med al information om bestillingen, så jeg ved hvilke dele, der skal bruges, med deres varenummer, antal, hjælpetekst, beskrivelse, og mål.(must have)	Givet, at jeg er sælger, og jeg står på "tilpasning af kundes carport"-siden, og jeg har fået indsat al data fra kundens ønsker, kan jeg trykke "Generer Stykliste" og modtage en liste med al information om de dele der skal bruges og antal.
<b>14.</b> Som sælger, når jeg bruger hjemmesiden til at generere kundens carport, så ved hjemmesiden om hvorvidt taget er med hældning eller ej, baseret på materialet for taget, så jeg kan sikre mig, at jeg får sat den hældning, kunden ønsker.(nice to have, da vi i første omgang skal have implementeret en med fladt tag)	Givet at jeg er kunde når jeg selv designer min carport vil jeg gerne have muligheden for at kunne vælge materialer til taget som kan have hældning eller ej.
<b>15.</b> Som kunde vil jeg kunne få højere pris baseret på min lokalitet fra FOG Lagerhus. (nice to have)	Givet at jeg er kunde og bestiller fra Jylland mens jeg er i Sjælland, vil jeg gerne have højere priser for carporten.
<b>16.</b> Som kunde vil jeg gerne kunne se en detaljeret beskrivelse af carporten når jeg har valgt den.(nice to have)	Givet at jeg som kunde har trykket på en carport vil jeg på den carports side kunne se en detaljeret beskrivelse af carporten.

<p><b>17.</b> Som kunde vil jeg gerne kunnen vælge om jeg hyrer FOG til at sætte carporten op for mig eller gør det selv.(must have)</p>	<p>Givet at jeg som kunde har valgt eller designet min egen carport vil jeg gerne have muligheden for at kunne bede Fog om at sætte den op for mig eller ej.</p>
<p><b>18.</b> Som kunde vil jeg kunne hente en monteringsvejledning, så jeg kan installere min brugerdefinerede carport, hvis jeg har betalt.(must have)</p>	<p>Givet at jeg har betalt for en brugerdefineret carport, vil jeg gerne kunne hente en plan til byggelsen af carporten, hvis jeg har betalt.</p>