

Semesterprojekt

Tjalfe Møller, Kristian Krog, David Carl og Kasper Breindal

29. maj, 2017

Indhold

Brugergrænseflade og design	3
Gennemgang af grænseflade	3
Tekniske detaljer	6
SQL Queries	8
Testing	10
Manuel Testing	10
Automatiseret Testing (JUnit)	10
Arbejdsmetoder	11
Værktøjer	11
Scrum	11
Agile development	12

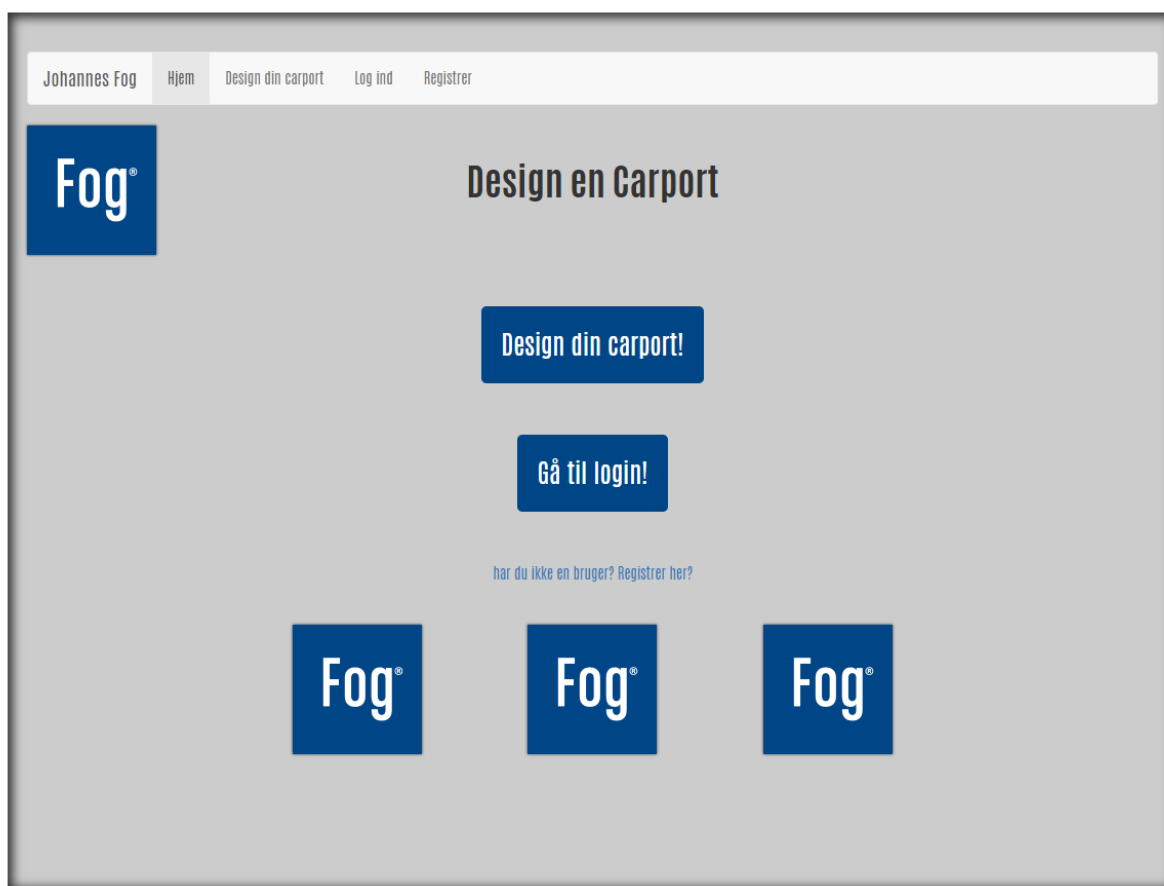
Brugergrænseflade og design

Af Kasper

Vores mål med programmet var at skabe en nemt og velfungerende måde at designe en carport på. Det skulle føles intuitivt, og ikke skabe forvirring når man arbejder i programmet. Programmet har været igennem flere forskellige stadier, før det nåede frem til vores endelige løsning.

Gennemgang af grænseflade

For brugere

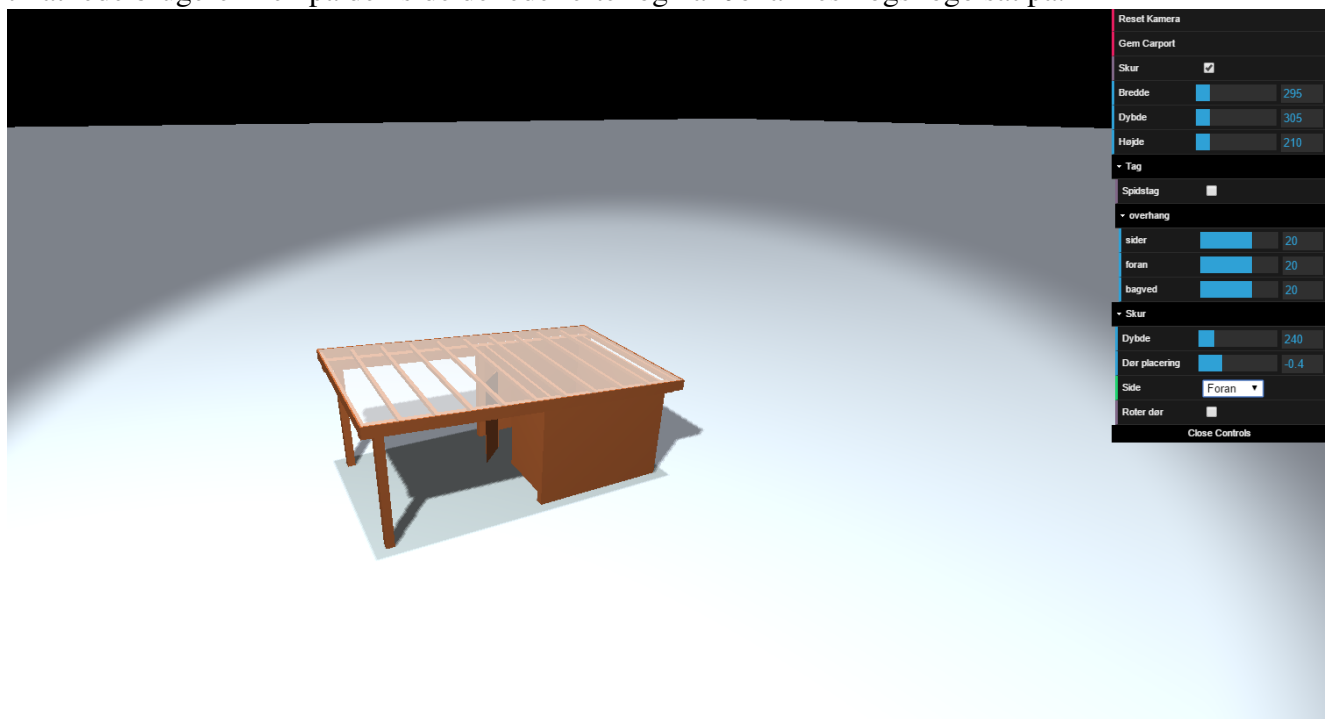


Det første der skal tages stilling til for brugeren af siden er, om man vil logge in, registrere sig, eller bare prøve at designe sin carport. Dette valg er bevidst stillet op for at tiltrække så mange potentielle kunder som muligt.

Kunderne vil ikke som det første på siden oprette sig. Derfor får de muligheden for at arbejde med de-

signværktøjet først, og gemme det senere.

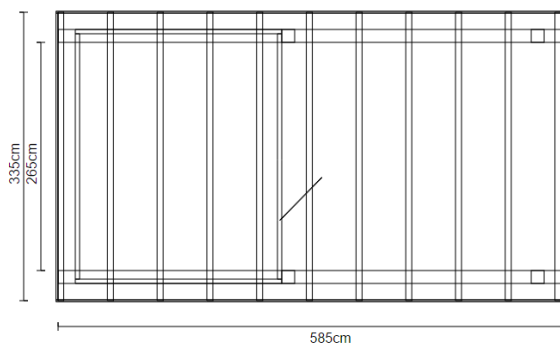
Designmæssigt har vi ikke haft mulighed for at snakke med productowner Johannes Fog, og derfor har vi implementeret et design som er simpelt og nemt at finde rundt i. Navigationsbaren i toppen kan bruges til at lede brugeren hen på den side de leder efter og har Johannes Fogs logo sat på.



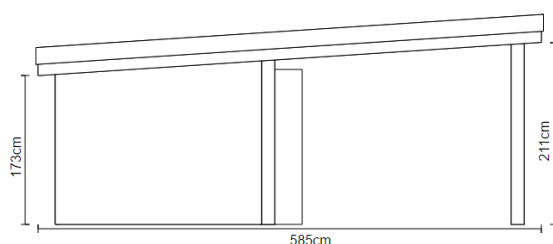
Ved tryk på ”design din carport” knappen ledes brugeren til vores 3D design program. Her kan brugeren køre sliders frem og tilbage, og vurdere hvordan deres carport skal se ud, og hvilke specifikke mål den skal have. Dette har den fordel at kunderne får en grafisk præsentation af hvordan deres carport kommer til at se ud, istedet for bare et billede af en tilfældig anden carport som firmaet engang. Når kunden føler at deres carport er som den skal være klikker de på ”gem” knappen, og bliver herefter ledt videre til plantegningen af deres carport.

Price:

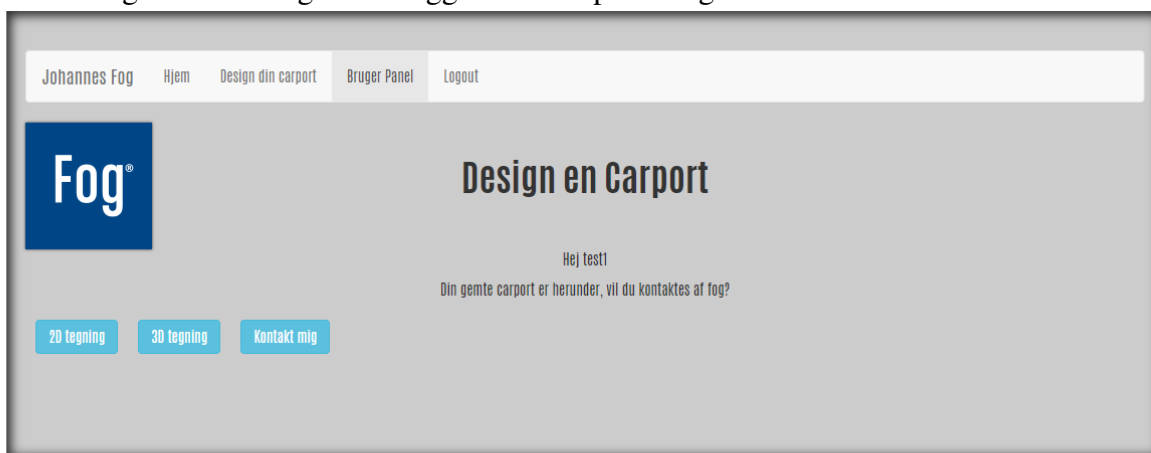
top:



side:



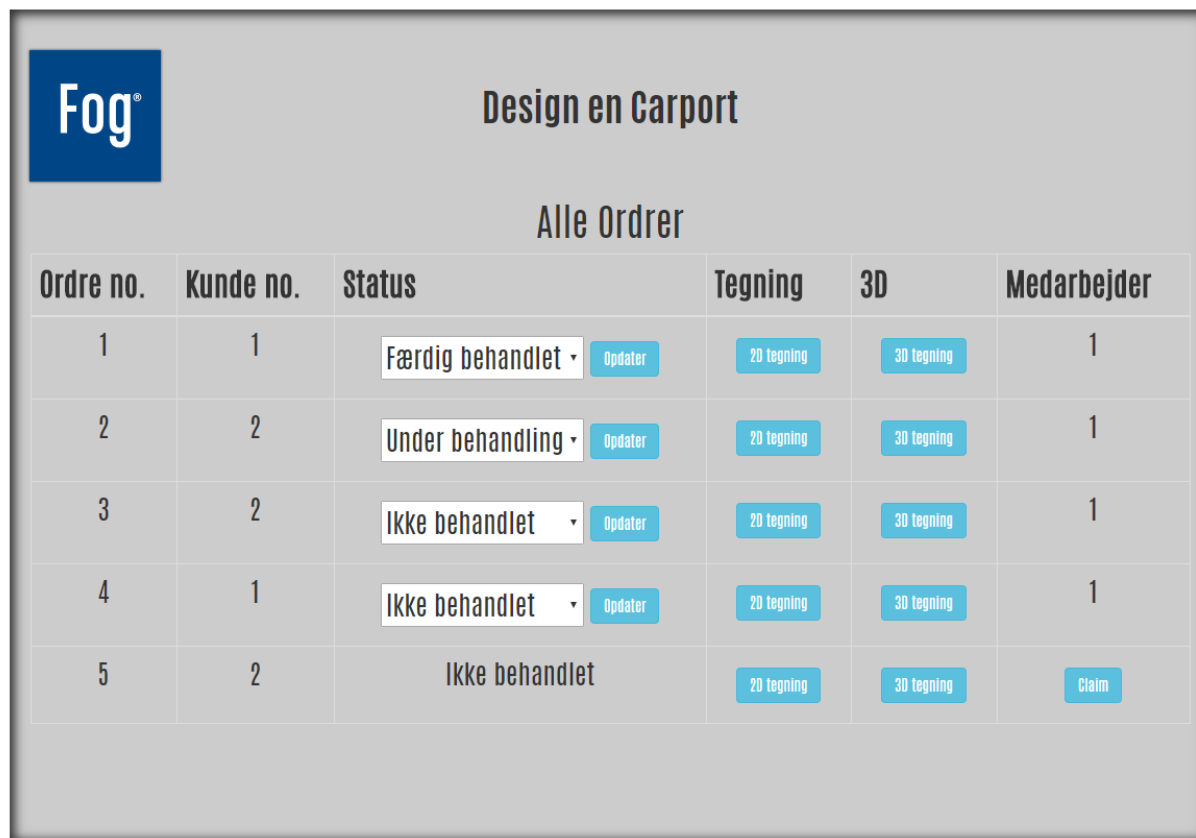
Her får kunden en 2D-plantegning af den carport de netop har designet i 3D-programmet. Tegningen har ikke alle mål sat på - dette er gemt til kunder der vælger at købe den carport de har designet. Samtidig med at tegningen er blevet genereret er carporten blevet gemt til brugeren, hvis man er logget ind. Hvis ikke, vil den bagefter bede dig om at logge in eller oprette dig.



Til slut kommer brugeren til brugerpanelet. Her kan de se den carport de har gemt, og kan trykke bestil, hvorefter ordren bliver sendt til databasen, til videre behandling af Fogs medarbejdere.

for Fogs medarbejdere

Fogs medarbejdere kan benytte dette samme system til behandling af ordrene. Hvis en medarbejder logger ind vil de blive taget til administratorpanelet. Her kan medarbejderne se status på de forskellige ordrer - om de er helt nye, under behandling af en kollega eller færdigbehandlet og klar til levering.



Ordre no.	Kunde no.	Status	Tegning	3D	Medarbejder
1	1	Færdig behandlet ▾ Opdater	2D tegning	3D tegning	1
2	2	Under behandling ▾ Opdater	2D tegning	3D tegning	1
3	2	Ikke behandlet ▾ Opdater	2D tegning	3D tegning	1
4	1	Ikke behandlet ▾ Opdater	2D tegning	3D tegning	1
5	2	Ikke behandlet	2D tegning	3D tegning	Claim

Tekniske detaljer

Websiderne i projektet er skrevet i JSP, CSS og JavaScript, og benytter sig af JSTL og eksterne libraries så som THREE.js, jQuery og Bootstrap til at forbedre udseendet af projektet og gøre projektet kompatibelt med flere forskellige browsere og styresystemer. Så vidt muligt bliver disse libraries hentet eksternt, via et Content-Delivery Network (CDN). Dette skaber den fordel at brugerne kan have disse pakker cached fra tidligere brug på andre websider i forvejen, og derfor ikke behøver at hente dem igen. Det betyder også at pakkerne ikke kommer til at ligge på firmaets server, og derfor kan lette datatrafikken på serveren, og derved køre mere stabilt.

Workflowet er udarbejdet efter at kunne indsættes i et hvert webside-design, og til at kunne omskrives til andre programmeringssprog.

Dette giver os som gruppe muligheden for at udvikle i det sprog vi kender til (Java), samtidig med at det kan omskrives til, for eksempel, en C# og ASP.Net baseret kodebase, hvilket er meget udbredt i Danmark.

SQL Queries

Af Kasper

Alle metoder til at behandle SQL statements til vore MySQL database findes i DataAccessObject-klassen. Vi benytter os af PreparedStatements for at gøre systemet hurtigere og mere stabilt, og for at sikre os mod SQL-injection angreb.

```
44
45  * @param username to get data about.
46  * @return User object containing found data.
47  * @throws SQLException if the sqlstatement is wrong or, the access to the
48  * database is wrong.
49  */
50 public User getUserByUsername(String username) throws SQLException {
51     User user = null;
52     PreparedStatement stmt = null;
53     try {
54         stmt = dbcon.getConnection().prepareStatement("SELECT * FROM users WHERE uname = ?;");
55         stmt.setString(1, username);
56         ResultSet rs = stmt.executeQuery();
57         if (rs.next()) {
58             int UID = rs.getInt("uid");
59             String usernameRetrieved = rs.getString("uname");
60             String passwordRetrieved = rs.getString("password");
61             String saltRetrieved = rs.getString("salt");
62             String emailRetrieved = rs.getString("email");
63             String userString = rs.getString("userstring");
64             String carportRetrieved = rs.getString("carport");
65
66             user = new User(UID, usernameRetrieved, passwordRetrieved, saltRetrieved, emailRetrieved, userString, carportRetrieved);
67             System.out.println(user.getUserName());
68         }
69     } finally {
70         try {
71             if (stmt != null) {
72                 stmt.close();
73             }
74         } catch (Exception e) {
75             //
76         }
77     }
78     return user;
79 }
```

PreparedStatements indsætter de strings der bliver givet med til metoden der hvor spørgsmålstegnene står. Dette er ligegyldigt hvad der står i den string der kommer med, derfor bliver SQL-injection umuliggjort her.

MySQL-Databasen er opbygget specifikt efter vores egne behov. Derfor er den ikke særlig fleksibel. Dog er tabellen med dele til carporten bygget op så både skruer, brædder, søm osv. kan være i samme tabel. dette er gjort ved at definere forskellige dele-id'er som markerer forskellige typer materialer.

DigitalOcean x

File Edit View Query Database Server Tools Scripting Help

Navigator: SQL File 3 users Database material x

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

PERFORMANCE

- Dashboard
- Performance Reports
- Performance Schema Setup

SCHEMAS

Filter objects

fogdb

- Tables
 - adminuser
 - material
 - orders
 - userdata
 - users
- Views
- Stored Procedures
- Functions

sys

Information: No object selected

SQL File 3

```
1 • SELECT * FROM fogdb.material;
```

Limit to 1000 rows

Result Grid

mno	type	price	name	qoh	size
2	skrue	400	Plastmo Tagskrue	50	200
3	træ	200	Spærtræ	45	1
4	skrue	100	Spærskrue	100	50
5	beslag	12	Tagspærbeslag	50	1
6	skrue	220	Hulbåndsskrue	100	20
7	træ	145	Tagrem	40	1
8	tag	200	Tagpap	900	1
9	tag	900	Danspær	20	1
10	træ	100	Sternbræt	90	1
11	træ	150	Stolpe	50	1
12	skrue	160	Stolpeskrue	50	50
13	skrue	160	Stolpebolt	50	20
14	træ	40	Skurbræt	700	1
15	skrue	200	Skursøm	60	200
16	træ	120	Løsholte	30	1
17	beslag	200	Hulbånd	30	1
*	NULL	NULL	NULL	NULL	NULL

material 1 x

Testing

Af Kasper

I softwareudvikling er det vigtigt at sikre sig, at det der bliver produceret også fungerer som det skal. Firmaer benytter software til både behandling af persondata, lønstyring og behandling af anden fortrolig data, og derfor er det vigtigt, at alt er sikkert og fungerer efter hensigten når det bliver sendt live.

Manuel Testing

Vi har alle i gruppen individuelt gået igennem alle funktioner i programmet, og sikret os at det hele fungerer. Dette har inkluderet at prøve at sende forkert data i input-felter, se sider, hvor man skal være logget ind som administrator, uden at være det og tjekke efter ødelagte links. Vi har derefter fikset alle fejl der blev fundet, for at levere et bedre produkt.

Automatiseret Testing (JUnit)

Vi har Unit Tests på de mest essentielle metoder i klassen DataAccessObject, da dette er den klasse der foretager de fleste vigtige metodekald.

Arbejdsmetoder

Af Kasper

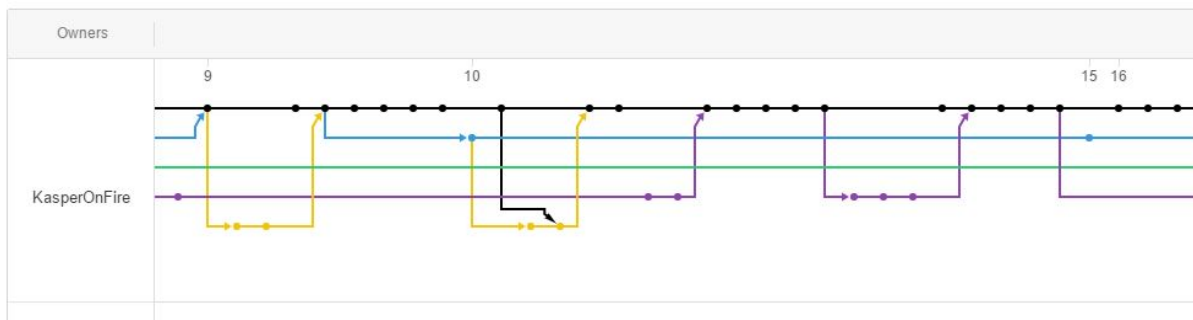
En del af opgaven var at følge Scrum og Agile principper til at få fremstillet projektet. Gruppen tog dette nogenlunde til sig, dog krævede det en tilvænningsperiode i starten af samarbejdet før alle var med på hvordan dette skulle udføres.

Vi har været 4 mand i gruppen, og selvom Scrum bygger på grupper af 6-8 mand, har dette har været en meget passende størrelse i forhold til uddellegering af opgaver, og så er man ikke for mange til møderne, som derfor bliver overkommelig længde. Derudover gør det mindre antal gruppemedlemmer også det, at det ofte er nemmere at komme til enighed i diskussioner, men at alles stemme bliver hørt, da hver mand jo tæller for 25%.

Vi har i vores gruppe aftalt at vi de fleste dage mødtes på skolen for at arbejde, for at sikre os at alle får foretaget det de skal, og det på den måde bliver nemmere at hjælpe hinanden med problemer man måtte møde undervejs.

Værktøjer

Til versionsstyring har vi benyttet os af Git og GitHub. Dette har fungeret til al tilfredsstillelse, og vi har sjældent haft merge conflicts. Gruppen har sigtet efter at arbejde efter feature-branching, altså at hver ny tilføjelse laves i sin egen branch, og derefter merges ind i master til det fungerende projekt.



Scrum

Vi har arbejdet efter scrum i dette projekt. Dette har inkluderet daily standup meetings, sprint planning, sprint reviews og sprint retrospective meetings. Dette projekt har været god øvelse i at arbejde efter scrum's principper. Især daily standups har været effektive, da det hurtigt giver et overblik over hvad gruppen arbejdede med dagen før, og hvad der skal arbejdes med denne dag.

Agile development

Agile development kom nemt til gruppemedlemmerne som en teknik, da alle fokuserede på at levere funktionel kode og at være produktive individuelt. Enkelte gange har problemer krævet at 2 satte sig sammen og par-programmerede for at løse dette.