michael due p., [CPH-MP447@CPHBUSINESS.DK](mailto:CPH-MP447@CPHBUSINESS.DK), MICH561D

j. cHRISTIAN rYGE, [cpH-JR221@CPHBUSINESS.DK](mailto:cpH-JR221@CPHBUSINESS.DK), rANGERrYGE

github link: <https://github.com/mich561d/ProjectFog>

weblink: ip:port:???

Cupcake Projekt Rapport

Gruppe: CRMD

Vinter 2018

Indholdsfortegnelse

[Indledning 2](#_Toc532196459)

[Baggrund 2](#_Toc532196460)

[Valg af teknologi 2](#_Toc532196461)

[Krav 2](#_Toc532196462)

[Overordnet beskrivelse af virksomheden 2](#_Toc532196463)

[Arbejdsgange der skal IT-støttes 3](#_Toc532196464)

[Scrum userstories 3](#_Toc532196465)

[Modeller og diagrammer 3](#_Toc532196466)

[Domæne model 3](#_Toc532196467)

[Klasse diagram 4](#_Toc532196468)

[EER diagram 4](#_Toc532196469)

[Navigationsdiagram 4](#_Toc532196470)

[Sekvens diagram 4](#_Toc532196471)

[Særlige forhold 5](#_Toc532196472)

[Udvalgte kodeeksempler 5](#_Toc532196473)

[Status på implantation 6](#_Toc532196474)

[Test 6](#_Toc532196475)

[Process 6](#_Toc532196476)

[Arbejdsprocessen faktuelt 6](#_Toc532196477)

[Arbejdsprocessen reflekteret 6](#_Toc532196478)

[Bilag 7](#_Toc532196479)

# Indledning

I følgende projekt skal der laves en hjemmeside for Johannes Fog, hvor på der kan bestilles en konfigurer bar carport efter en brugers egne valg og ønsker til materialer samt udformning. Før end en bruger kan bestille skal personen oprettes i databasen, hvorfor vi har lavet en mulighed for at man kan registrere sig på siden. Herefter kan der logges ind på siden. Når en bruger er logget ind, kan de se tidligere ordre samt ændre deres oplysninger.

Vi har også lavet en administrator side, hvori der kan ændres i produkters data, pris, beskrivelse osv. Ydermere skal en administrator kunne oprette nye administratorer og behandle brugeres ordre.

## Baggrund

**Det typiske der skal med for at forklare projektet er:**

Johannes Fog er et byggemarked der samtidigt også består af et Bolig og Designhus. Johannes Fog dækker Sjælland med deres ni byggemarkeder, hvoraf det ene befinder sig i Vordingborg og de øvrige i Nordsjælland. De beskæftiger sig primært med træ og tilbehør hertil, men de har også stor ekspertise i salg af jern, stål og andre metaller. Johannes Fog kan levere til alle slags opgaver og fokusere på et bredt udvalg i størrelser og kvaliteter. Samtidigt kan de også levere værktøj til alt slags arbejde.

## Valg af teknologi

Vi har valgt at skrive programmet i Java. Derfor faldt valget på Netbeans, idet programmet effektivt håndterer de valgte programmeringssprog og understøtter Apache Tomcat, hvilket er et godt serverværktøj. Til opbygningen af databasen har vi valgt MySQL, da deres Workbench er et godt udviklet SQL-program.

|  |  |
| --- | --- |
| **Programmer:** | **Sprog:** |
| * Netbeans 8.2 * MySQL Workbench 8.0 * Java & Java SE (JDBC) 181 * Apache Tomcat 8.0.27.0 | * Java * HTML * CSS * MySQL 14.14 |

# Krav

Der er to dele i dette afsnit:

* hvad er firmaets håb med dette system (hvad er deres vision for systemet eller hvilken værdi er det jeres system skal tilføre deres virksomhed).
* scrum userstories (product backlog)

## Overordnet beskrivelse af virksomheden

Hvad laver virksomheden? Grundlæggende skal dette afsnit besvare de spørgsmål som i har besvaret med SWOT analyse og interessant analyse tidligere.

## Arbejdsgange der skal IT-støttes

Dels skal afsnittet beskrive de overordnede arbejdsgange før og efter IT systemet. Gerne beskrevet med “as-is” og “to-be” som aktivitets diagrammerne.

## Scrum userstories

Dette afsnit skal beskrive de user-stories der er aftalt med product-owner. Det er vigtigt for hver user-story at man kan spore det tilbage til aktivitetsdiagrammerne fra forrige side, og at I har en håndfuld userstories som er lavet fuldt ud, dvs:

* der er beskrevet how-to-demo
* der er brudt ned i tasks
* der er lavet et estimat
* den enkelte story følger INVEST princippet
* at det er klart hvordan den tilfører værdi

Den fulde produkt backlog kan ligge som appendix.

# Modeller og diagrammer

## Domæne model

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 Domæne eller 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)

## Klasse diagram

## EER 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 Domæne eller 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)

## Navigationsdiagram

Det som brugeren oplever er en række websider hvor man kan indtaste oplysninger gå videre til andre sider. I større systemer kan det være svært at bevare overblikket over hvilke sider der er, og hvordan man kommer fra den ene til den anden. Navigationsdiagrammet er beregnet på at vise dette på en mere overskuelig måde. Som led i beskrivelsen af navigationsdiagrammet skal følgende med:

* Oversigts diagrammet. Hvis det bliver for stort må man dele det op. Men det er vigtigt at der er et overordnet diagram.
* Hvis man har benyttet sig af en “fælles navigations bar” i toppen af alle sider skal man forklare det.
* Hvis nogle sider kun kan nåes af nogle brugere (dem der har konto, dem der er logget ind, dem der arbejder i butikken,…), så skal det fremgå.
* Navne på jsp sider skal fremgå, og hvilke servlet der bringer en fra den ene side til den næste.

## Sekvens diagram

De fleste programmører kan læse de enkelte metoder i et program, mens det kan være svært at skabe sig et overblik over hvordan programmet virker på overordnet plan. Et sekvens diagram bruges til at vise hvordan et typisk forløb foregår, eller til at vise et særligt svært særtilfælde. Man kan aldrig dokumentere hele programmet med sekvensdiagrammer, man vælger altid nogle interessante eksempler.

Et eksempel på et typisk forløb kunne være at brugeren præsenteres for indkøbssiden. Her skal der vises følgende:

* Selve diagrammet, startende med jsp-siden eller servlet.
* Brug de rigtige klassenavne og metode navne
* Undlad argumenter til metoderne

I forklaringen til diagrammet skal du særligt lægge vægt at beskrive hvilke grene af if-sætninger der er brugt i de enkelte metoder.

# 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.

# 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.

# Status på implantation

Dette afsnit skal liste hvor langt man er nået med implementationen. Typiske ting man kan have sprunget over er:

* Man har ikke nået at lave alle de jsp sider man har med i navigationsdiagrammet.
* Man har ikke nået at lave alle CRUD metoderne til alle tabellerne
* Man har ikke fået stylet sine sider
* Man har fundet en fejl “i sidste øjeblik”, men har ikke haft tid til at rette det. - F.eks. at man har brugt session forkert sådan at man på en af siderne kan komme ind uden at være logget in.
* tests der fejler på afleveringstidspunkt
* … andre mangler

# Test

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.

# 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.

## 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.

## 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

# Bilag