CUPCAKE WEBSHOP

af

Morten Bendeke

cph-mb809@cphbusiness.dk

https://github.com/coerth/CupcakeV2

og

Denis Pedersen

cph-dp153@cphbusiness.dk

https://github.com/coerth/CupcakeV2

CPH-Business
Lyngby

d. 21/4-2022

Indholdsfortegnelse

Indledning

Olsker Cupcakes fra Bornholm har hyret os til at lave en hjemmeside til deres forretning. Hjemmesiden skal fungere som en platform hvorfra kunden bestiller og betaler for cupcakes ud fra de mulige kombinationer af top og bund som butikken stiller til rådighed. Herefter kan kunden afhente sine cupcakes i butikken i Olsker.

Kunden skal kunne oprette en konto hvorfra der kan betales og gemmes ordre samt logges ind ved senere besøg. Fra denne profil skal kunden også kunne set sine valgte ordrelinjer og den samlede pris i en indkøbskurv. Kunden skal have mulighed for at fjerne ordrer fra indkøbskurven efter de er tilføjet. Når der er logget på skal kundens e-mail fremgå af hjemmesiden.

Administratoren af hjemmesiden skal kunne indsætte beløb direkte på en kundes konto i MySQL.

Administratoren skal kunne se alle kunder og ordrer i systemet samt fjerne ordrer igen. Ligesom kunden skal administratorens e-mail også fremgå af hjemmesiden efter der er logget ind.

Vi gik til opgaven med ønsket om at lave MVP (*minimum viable product*) med klart fokus på *backend*. Først når backend var helt færdiggjort ville vi kigge på at *style* hjemmesiden med CSS osv. De funktionelle krav vi blev stillet var i fokus

Morten Bendeke
cph-mb809@cphbusiness.dk
dat-0921b

Denis Altin Pedersen
cph-dp153@cphbusiness.dk
dat-0921b

Baggrund

Olsker Cupcakes er et dybdeøkologisk iværksættereventyr fra Bornholm som har nået en størrelse som gør, at de har behov for en hjemmeside der kan hjælpe dem med salget af cupcakes. Virksomheden lever stort på, at kunderne selv kan "bygge" en cupcake ud fra de forskellige bunde og toppe som der stilles til rådighed.

Der er stillet en simpel mock-up til rådighed og en række funktionskrav som virksomheden forventer hjemmesiden skal være i stand til.

Krav

Firmaet forventer en funktionel hjemmeside hvorfra kunden kan:

- Bestille og betale for cupcakes med en valgfri bund og top som kan afhentes i butikken.
- Oprette en profil hvorfor der kan betales og gemmes en ordre.
- Efter login skal kunden e-mail fremgå af hjemmesiden.
- Se valgte ordrelinjer i en indkøbskurv hvoraf der fremgår den samlede pris.
- Slette ordrer fra indkøbskurven.

Derudover forventes det at en administrator kan følgende:

- Indsætte beløb på en kundes konto direkte i MySQL.
- Se alle ordrer i systemet.
- Fjerne ordrer fra systemet.
- Se alle kunder og deres eventuelle ordrer.
- Ligesom for kunden skal administratorens e-mailadresse fremgå af hjemmesiden efter der er logget på.

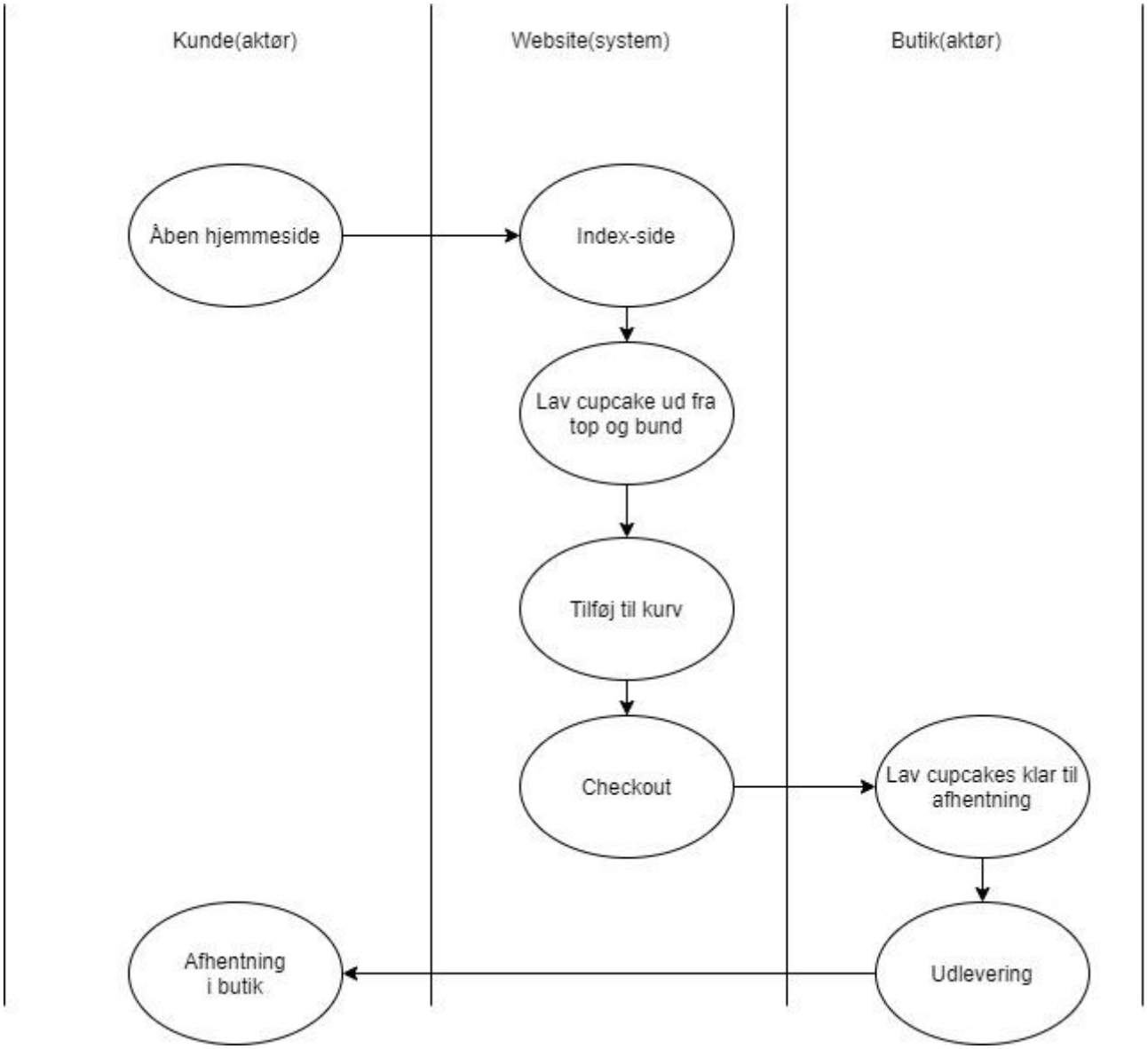
Valg af teknologi

Vi har, for at opfylde alle kravene, benyttet os af følgende programmer:

- Intellij 2021.3.1 Ultimate Edition til JAVA, CSS og JSP.
- MySQL Workbench 8.0 CE til at håndtere vores database og ER-diagram.
- Tomcat 9.0.601 som server/servlet.
- Github/Gitbash til arbejdet i gruppen.
- Draw.io til mockup og diagrammer.

Aktivitetsdiagram

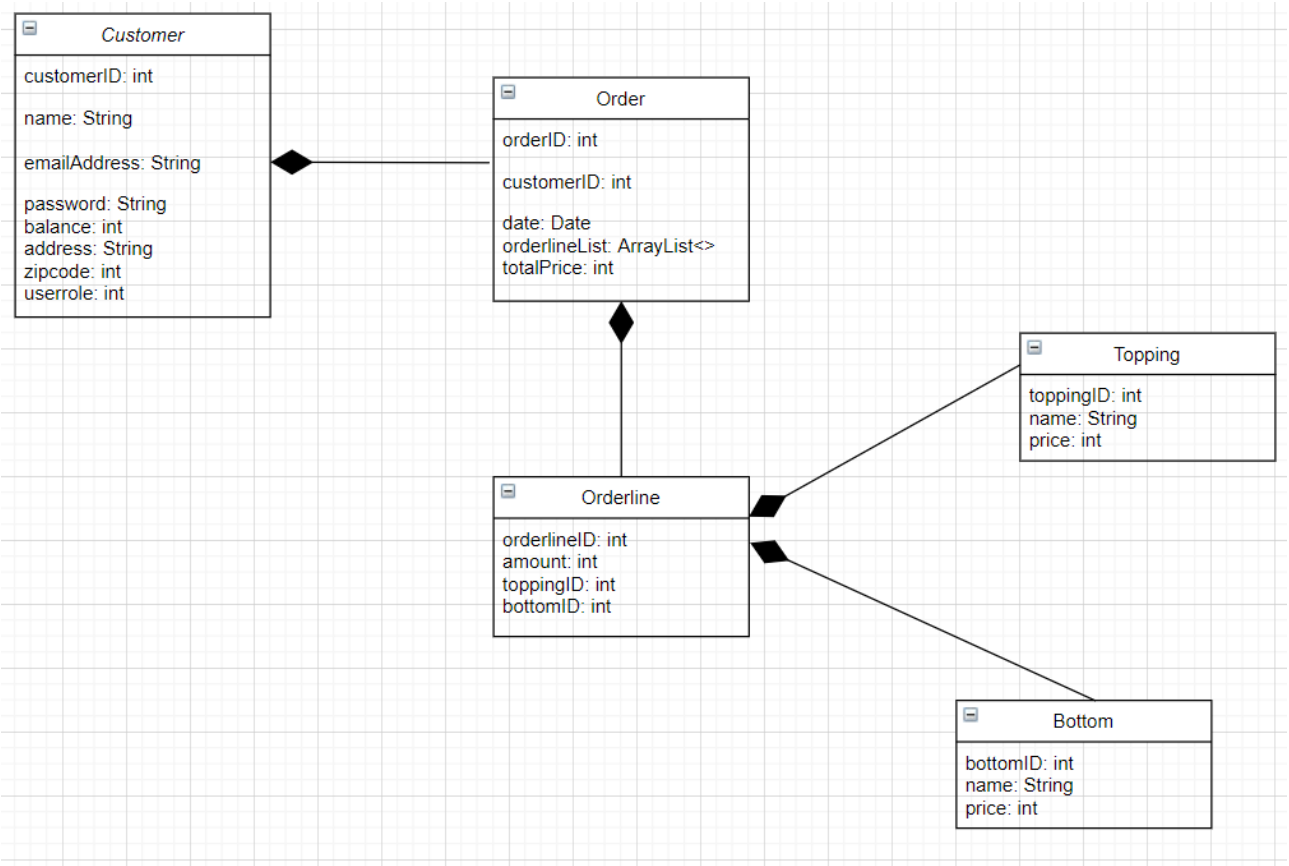
Vores aktivitetsdiagram viser TO-BE. Her er det meningen, at kunden bestiller sin cupcake ud fra de toppe og bunde som er stillet til rådighed. Herefter kan kunden afhente sine lækre cupcakes i butikken på Bornholm.



Morten Bendeke
cph-mb809@cphbusiness.dk
dat-0921b
Denis Altin Pedersen
cph-dp153@cphbusiness.dk
dat-0921b

Domænemodel

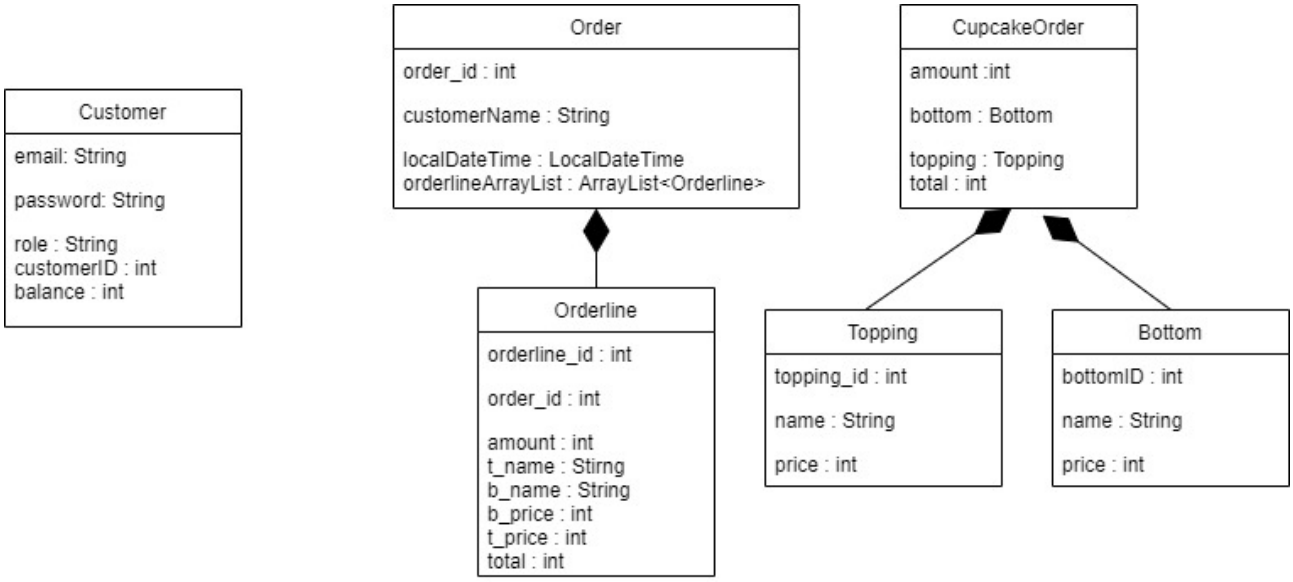
Vi startede med at lave en domænemodel med de variabler og objekter som vi forventede ville blive nødvendige på siden. Den første model så således ud:



Efter vi startede på projektet og skulle håndtere både ordrehistorik og indkøbskurv måtte vi adskille indkøbskurvens ordrelinjer fra selve cupcake-objektet. Så en cupcakeOrder er en linje i indkøbskurven som har den nødvendige information der skal bruges for at bekræfte en ordre. I praksis er både vores Order og Orderline DTOer, da vi sammensætter relevant information fra databasen som skal med i ordrehistorikken. Al dette medførte at vores domænemodel endte med at se sådan ud:

Morten Bendeke
cph-mb809@cphbusiness.dk
dat-0921b

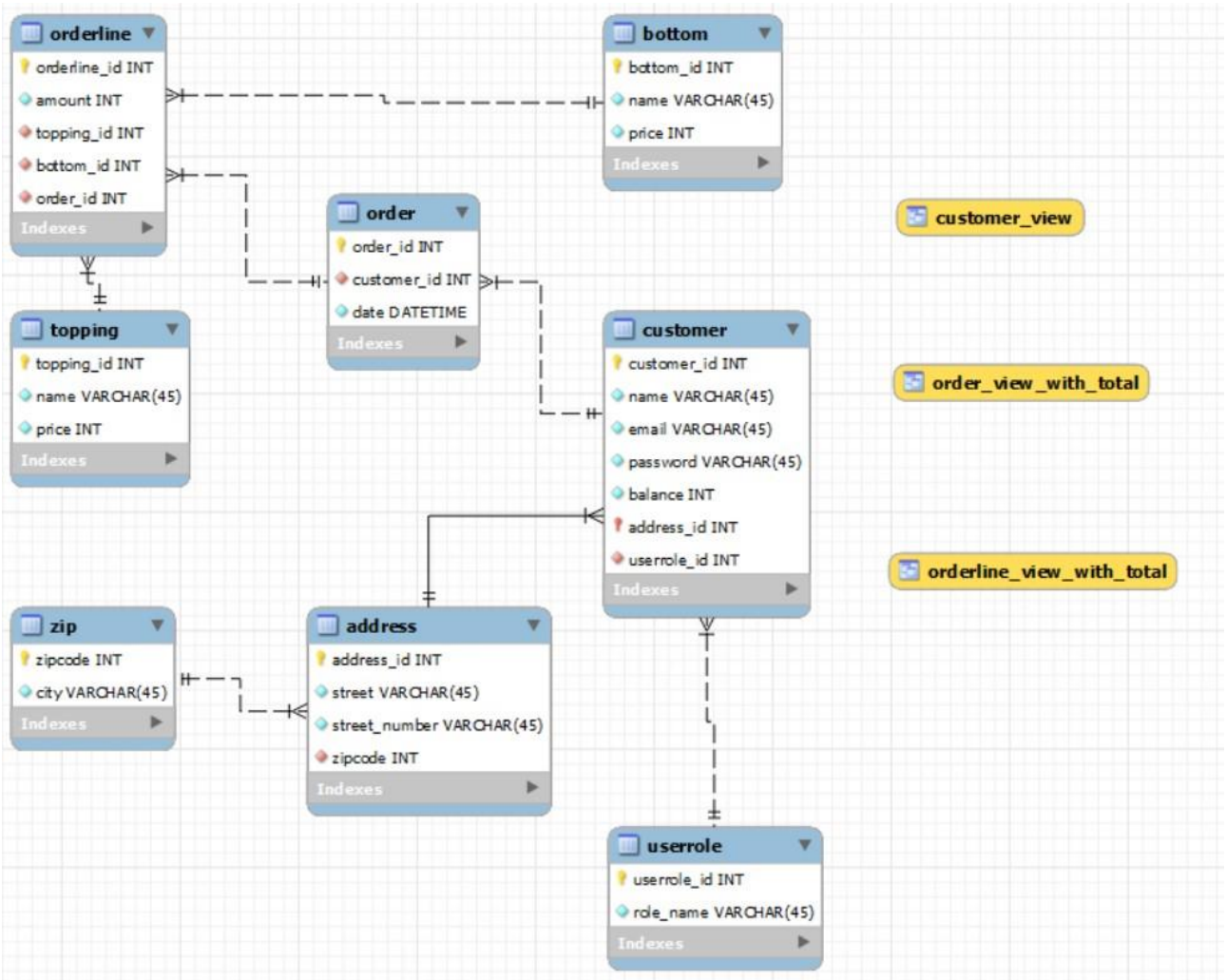
Denis Altin Pedersen
cph-dp153@cphbusiness.dk
dat-0921b



Morten Bendeke
cph-mb809@cphbusiness.dk
dat-0921b
Denis Altin Pedersen
cph-dp153@cphbusiness.dk
dat-0921b

ER-diagram

Vi havde en klar intention om at normalisere ud fra tredje normalform. Derfor ses der en zip-tabel separat fra en adresse. Man kunne argumentere for, at de begge kunne være en linje i Customer (dette blev også anbefalet fra underviserens side), da der ikke er levering, men ud fra vores normalisering ville denne funktion let kunne implementeres i systemet.

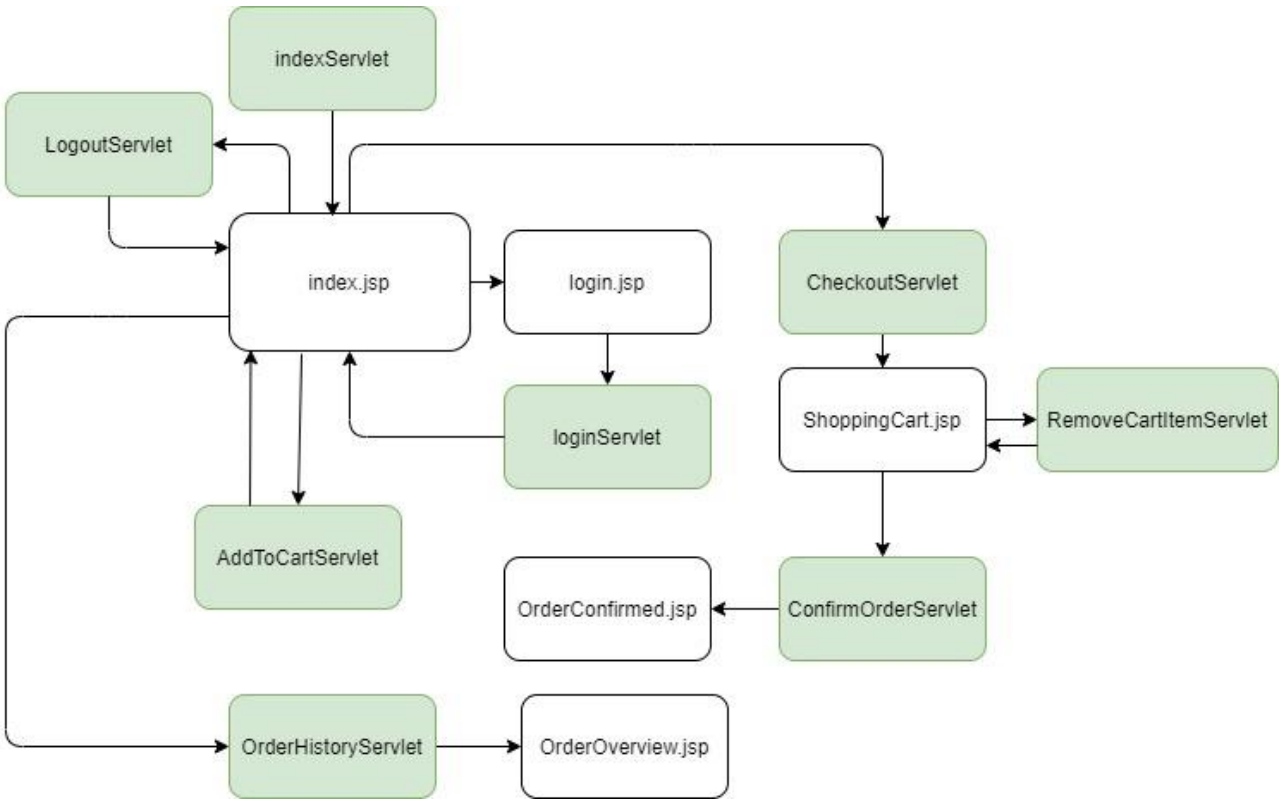


Morten Bendeke
cph-mb809@cphbusiness.dk
dat-0921b

Denis Altin Pedersen
cph-dp153@cphbusiness.dk
dat-0921b

Navigationsdiagram

Vores side har en navigationsbar i toppen, som er ens for alle jsp-sider. Herfra kan brugeren tilgå login, indkøbskurv og se ordrehistorik. Ordrehistorikken har forskelligt udseende og indhold alt efter om brugeren er logget ind som admin eller kunde.



Morten Bendeke
cph-mb809@cphbusiness.dk
dat-0921b

Denis Altin Pedersen
cph-dp153@cphbusiness.dk
dat-0921b

Særlige forhold

Under design og implementering har vi truffet følgende valg:

- Bottom- og topping-valgmuligheder bliver gemt på context/applications-scopet. Det er gjort, da vi finder disse valg relevante for samtlige brugere af siden.
- De valgte bunde og toppings (cupcakes) samt antal bliver gemt på sessions-scopet i en ArrayListe af CupcakeOrders. Det er i praksis vores indkøbskurv. De bliver gemt på sessions-scopet, da de ikke er relevante efter kunden besøg på siden.
- Efter bekræftet ordre bliver ArrayListen kørt igennem og hermed laves ordre og ordrelinjer i databasen. Herefter bliver ArrayListen nulstillet, da indkøbskurven nu er tom igen.
- Vi har valgt af oprette en cupcake-bruger på SQL-serveren som vi benytter til at lave database-forbindelsen.
- Efter brugeren er logget på tilknyttes denne session-scopet vha. et CustomerObjekt.
- Der er forsøgt at begrænse kundens muligheder for fejl gennem vores valg af en drop-down-liste. Hermed kan brugeren ikke komme til at skrive et tal eller andet hvor der kræves en topping osv. Der kan kun vælges op til 100 af hver cupcake. 100 er valgt da vi ikke er bekendte med bageriets begrænsninger, det kan sættes op hvis behovet opstår.
- Der er valgt at omdirigere til en fejl-side hvis brugeren forsøger at foretage handlinger der ikke er tilladt. Vi forsøger at håndtere null-pointer-exceptions sådan at de ikke har indflydelse på kundens navigering på siden, det gør vi bl.a. ved at kundens kurv kan være tom men kunden stadigvæk kan tilgå ShoppingCart.jsp samt at man kan gå til kurv uden at være logget ind.

Morten Bendeke
cph-mb809@cphbusiness.dk
dat-0921b
Denis Altin Pedersen
cph-dp153@cphbusiness.dk
dat-0921b

Status på implementation

Ud fra de krav vi blev stillet fik vi lavet følgende:

- Login/logout-funktion både for administrator og kunde.
- Vis email-adresse efter der er logget på.
- Lav en cupcake ud fra en liste af tilgængelige toppe og bunde og læg denne cupcake i en kurv.
- Kunden kan se ordrelinjer og ordrehistorik og samlet pris både og efter bestilling.
- Kunden kan fjerne ordre fra sin indkøbskurv inden ordren bekræftes.
- Administrator kan indsætte beløb direkte i databasen vha. MySQL.
- Administrator kan se ordrer for alle kunder på hjemmesiden.

Vi nåede desværre ikke helt i mål med følgende funktioner:

- Opret bruger-funktion → da vi havde valgt at normalisere ud fra tredje normalform, trods Jons gode råd om at undlade dette, blev opret-bruger-funktionen en rodet omgang, som vi desværre ikke nåede i mål med.
- Styling → Vi havde klare planer om at få lavet mere på frontend, men virkeligheden kom i vejen og vi havde ikke den fornødne tid.
- Administrator kan ikke se alle kunderne men kun deres ordrer → Vi havde ikke læst opgavebeskrivelsen ordentligt.
- Testing → Vi har ikke haft mulighed/tid til at lave integrations-test og unit-testing.

Morten Bendeke
cph-mb809@cphbusiness.dk
dat-0921b

Denis Altin Pedersen
cph-dp153@cphbusiness.dk
dat-0921b

Proces

Vi har i gruppen gået ud fra princippet om MVP (*minimum viable product*), og har først og fremmest fokuseret på *backend*. Det med farver og striber måtte vente til vi havde mere tid, hvilket vi desværre ikke fik i arbejdsprocessen, da der var rigeligt arbejde i de krav der blev stillet til funktionaliteten af vores hjemmeside.

Vi startede hver arbejdsdag med at sætte realistiske minimums-mål for dagen. Disse bestod både af mindre opgaver (opsætning af github, småændringer i JAVA-delen osv.) samt større opgaver (mappers, database osv.). Alt i alt var meningen med vores mål for dagen at gøre arbejdsprocessen mere overskuelig. Et eksempel på en dagsorden tidligt i processen:

05/04-2022
Mål for dagen:

1. Synkroniser i git og få refaktoreret kode + ændringer i databasens views. ✓
2. Lav mappers til cupcake-bunde og toppe ✓
3. Lav index til at init mappers ✓
4. Få vist data fra databasen vha. vores mappers. ✓
5. Lav login og opret bruger-funktionen i backend. ✓
6. Lav flere mappers og arraylister som vi kan få ud. ✓

Når vi så havde klaret en opgave satte vi et lille flueben og kunne med god samvittighed gå videre til næste opgave eller tilføje flere opgaver hvis der var mere tid.

Vi arbejdede i git, hvilket medførte et par bump på vejen i processen, men også viste sig som det fremragende arbejdsredskab det kan være. Da vi var godt i gang med projektet fik vi en lille forskrækkelse, da der opstod problemer med merge/rebasing. Problemet opstod, da vi havde fået en Cart.jsp lagt på git, som vi ikke formåede at få fjernet igen. Efter at have kæmpet med problemet i noget tid valgte vi, for at spare yderlige tid og tårer, at lave et nyt repository, hvorfra vi kunne fortsætte arbejdet. Et cupcake2.0-repository.

Da vi mødtes fysisk hver dag var der rig mulighed for at snakke sammen undervejs og støtte hinanden i de arbejdsopgaver vi hver især arbejdede på.

Vi var to i gruppen hvilket måske satte en stor arbejdsbyrde og tidspres på os begge? Havde en gruppe på 3 været mere passende til denne opgave? Det positive ved at være to var at vi fik rig mulighed for at få kodet hver især. Det gav en god indsigt i koden og der blev ikke lavet noget uden vi begge havde overblik over hvad der foregik. Derudover valgte vi, trods vores kære undervisers gode råd om det modsatte, at gå ud fra tredje normalform da vi normaliserede databasen. Det medførte en del ekstra arbejde som var svært at få det fulde

overblik over. Vi brugte lang, lang, lang tid på at få modal-popups til at fungere, men måtte kapitulere trods støtte fra undervisere, tutorer og medstuderende.

Til næste gang tager vi følgende overvejelser med:

- Læs opgavebeskrivelsen 100 % igennem.
- Hurtigere beslutning om at "gå videre", så vi ikke bruger for meget tid på at rende panden imod en mur. Det kunne have sparet os for en del udfordringer, hvis vi havde valgt at droppe modal-popups tidligere.
- Lyt til Jon når han siger at normalisering på anden eller første normalform er fint nok til opgaven.
- Vi brugte førstedagen på mock-up og diagrammer, hvilket gav en rigtig god platform for det videre arbejde.
- Git er et godt redskab, og vi har i processen haft god brug af det. Vi skal dog blive bedre til micro-commits og at squashe oftere.