Cupcake

af Daniel, Marc, Marcus, og Lasse

30/03-2020

Daniel Poulsen

Cph-Email: Cph-dp127@cphbusiness.dk

Github: Dane1998

Marc Ekström

Cph-Email: Cph-me206@cphbusiness.dk

Github: GGGE99

Marcus Jensen

Cph-Email: Cph-mj757@cphbusiness.dk

Github: MarcusJyl

Lasse Birk

Cph-Email: Cph-lb296@cphbusiness.dk

Github: LasseBirk96

Indledning

Systemet er lavet til at et bageri eller en ”cupcake cafe” kan modtage ordrer via en online webshop. Systemet kan bruges af to forskellige typer af brugere, kunder og administratorer.

Kunden kan oprette en bruger som han eller hun skal bruge for at gennemføre en ordrer og for at gemme diverse ordre til senere brug. Derefter kan kunden bestille skræddersyet cupcakes med deres eget valg af bund og topping, som senere hen skal kunne hentes i butikken. Kunden kan bruge den samme bruger hver gang de vil handle hos butikken.

Kunden kan også se deres bestilling i indkøbskurven for at få et overblik over deres bestilling. I indkøbskurven har kunden muligheden for at se alle bestilte cupcakes, den samlede pris for ordren, og have evnen til at slette en cupcake i indkøbskurven.

Da systemet ikke kan håndtere eksterne betalingsmetoder fra brugeren, er det nødvendigt at kunne have adgang til kundens konto og dermed saldo via systemet.

Administratoren kan derfor indsætte et beløb direkte til en kundes konto.

Udover det kan administratoren også se alle ordre og alle kunder, samtidigt med at han eller hun kan slette kunder og ordre, dermed har administratoren styr på alt information i systemet.

Teknologi

Systemet er blevet kodet i Java via Intellij IDEA 2019 3.3 og benytter sig af SQL workbench og JDBC som database. Systemet benytter sig også af build-automatiserings værktøjet Maven.

Krav

Det er værd at antage at firmaet har valgt at investere i en online webshop for at nå ud til flere kunder end de ellers havde haft mulighed for ved blot at have en fysisk butik. Ved at kunne modtage ordrer online kan butikken modtage flere ordrer end før, og samtidigt skaber det

også et bedre image omkring butikken at have deres egen hjemmeside, da det for dem til at fremstå mere professionelt.

Diagrammer

Domæne model

Domæne modellen over olsker cupcakes har to to aktører, kunder og admin. Kunder kan se deres kurv, hvori de kan slette en vare som de har lagt i kurven eller ændre antallet af varer lagt i kurven. Kunden kan også bestilles cupcakes, hvoraf en cupcake består af en topping og en bund. Cupcaken har en 1-1 relation, da der kun kan være en topping og en bund per cupcake, men der kan være 1-* cupcakes i en ordre. Til sidst har kunden også muligheden for at logge ud af systemet.
Administratoren har mulighed for at se to forskellige oversigter: Kundeoversigt og ordreoversigt. I kundeoversigten kan administratoren se alle oplysninger omkring kunder registreret i systemet herunder deres kundenummer, navn, adresse og balance. Her kan administratoren også slette en kunde. I ordreoversigten kan administratoren se alle ordrer og deres individuelle ordrelinjer med cupcakes. Til sidst kan administratoren ændre en kundes password.

ER Diagram

Databasen Cupcakes er lavet efter 3. normaliseringsprincippet hvor alle tabeller har relationer til hinanden i form af primary keys. I databasen har alle tabeller en til mange relationer med undtagelse af "cupcakes"-tabellen, som holder på de individuelle ordrelinjer med antal cupcakes, som kun kan have en topping og en bund som attribut. Tabellen cupcakes er blevet implementeret sådan at der er sikret at der kun er en ordrelinje per ordre i tabellen ordrer, så at alle ordrelinjer med cupcakes skal hentes fra cupcakes tabellen for at sikre at det 3. normaliseringsprincip bliver fulgt.

Navigationsdiagram:

Indexsiden indeholder login og via et pop up vindue kan du oprette dig som bruger. Når man som bruger er logget ind, har du en navigation bar der går igen på alle sider, det er hovedsageligt den der benyttes til at navigere rundt. Admin siden fremgår kun via en bruger med en administratorrolle, fra administrator siden kan kundeliste og ordre ses.
Fra kurv, færdiggøres bestillingen og du bliver ført tilbage til index.
Sådan så vores diagram ud før vi startede mens som vi arbejdede på projektet ændre vi i hvilke sider der var den dette kan ses på navigationsdiagram 2 i bilag her er der en række af sider der er blevet fjernet fra vores originale ide og nogle sider har ændret navn. Vi har dog stadig funktionaliteten fra alle de sider der var tænkt i starten. Dette har vi da vi har valgt at lægge flere af siderne sammen til en side i indexen da vi følte at de var for små til at skulle have sin egen side.

Sekvensdiagrammer

Sekvensdiagram for bestilling

Brugeren, i dette tilfælde kunde, vælger topping og bund på index.jsp siden og trykker på "Add". De variabler som brugeren vælger, sendes videre til Basket klassen, som det første tjekker ved en if sætning om basket i session er lig null. Hvis basket er null laver den en ny Arrayliste som kan holde på de cupcakes som bliver sendt med eller fortsætter den. Herefter opretter den en ny temp Cupcake og sender den videre til sessionAttribute Basket og returner index.jsp. Herefter, for at kunne bestille, bliver kunden videresendt til kurv.jsp hvor kunden kan se og gennemføre deres bestilling. Den sender en execute() videre til klassen Bestil som tjekker om brugeren er logget ind, ellers henter den userID og money og sletter kurven på index.jsp. Samtidig sender den update til databasen Cupcake, ved hjælp af orderMapper. Her sender den createOrder() til at ændre i ordrer tabellen, createOrderLine() og tager hver cupcake og splitter den op i top, bund og antal og updateMoney() for at opdatere brugerens balance i tabellen Money. Dette bliver sendt ind som en executeUpdate med SQL statements og sender oplysninger tilbage med resultset i form af en ny ordre med et ordre_id. Efter den har oprettet en ny ordre forwarder den brugeren til index.jsp.

Sekvensdiagram for login

Her indtaster brugeren en email og et password som bliver sendt fra index siden til Login classen hvor de bliver tage ud af requestScopet. Her bliver mailen og passwordet så sendt videre til login metoden i vores LoginFacade. Alt denne metode gør er at kalde en anden metode i UserMapper classen som også hedder login. Henter id, rolle og password ned for den mail der er blevet indtastet. Herefter tjekkes der om den indtastet kode er den samme som den fra databasen hvis den er det vi der laves en ny user i java som så returneres. Hvis koden derimod er forkert vil der her returneres null og hvis det er en mail som ikke kan findes i databasen vil det blive returneret en user med rollen ukendtMail. Hvis alt er godt som det skulle og man er blevet logget ind vil brugeren bliver returneret til index siden hvor deres mail nu kan ses i navbar sammen med en logout knap og for admins vil der også være en admin knap.

Særlige forhold

I sessionen bliver det gemmes kurven og information om bruger så bruges til at bestemme hvad der skal visse på siden. Her til er der lavet en metode der sætter bruger rollen til "ukendtEmail" og vis index siden ser at rollen er det vil den lave en pop up hvor man kan oprette sige. Hvis rollen derimod er "employee" så vil der være en admin knap i nav baren som ikke vil være der hvis rollen er "customer".

Vi har ikke rigtigt lavet validering af brugerinput men vi har gjort sådan at input felterne i vores html stemmer overens med de typer af variabler der skal bruges så brugerne ikke kan taste en værdig ind som ikke kan bruges.

Til login er det gjort sådan at hvis man indtaster en email som ikke kan findes i databasen så vil det sende dig til en side hvor du kan oprette en brugere. Hvis man indtaster en forkert koden vil den derimod ikke gøre noget: Siden vil bare blive på index siden og slette det man har indtastet i login felterne.

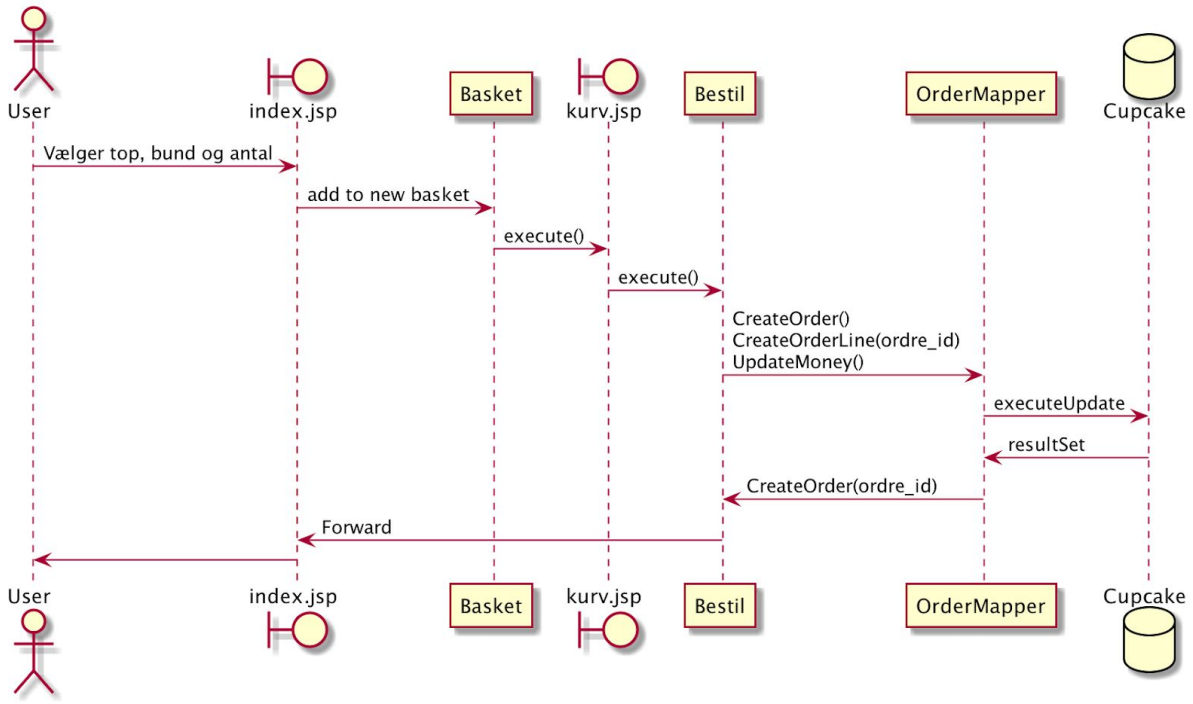
Status på implementation

Der er ikke blevet lavet meget i forhold til exceptions, der er blevet brugt det fra skabelonen der blev udleveret til os. Det den gør er den sender en til Login.jsp når der kommer en fejl menne denne side har vi ikke mere så man vil få en 404 error hvis der sker en fejl.
Der er endt med at være færre jsp sider end der var planlagt dette er gjort fordi vi fandt ud af at vi ville ligge login og oprettelses siden som en del af headeren. der er også kommet en ekstra side hvor man kan se order linjerne det er gjort da vi ikke kunne finde en pæn måde at vise alt information om en ordre på en side. Der mangler også en metode til at lave nye admin brugere på, men grundet tidsmangel er dette ikke blevet lavet.

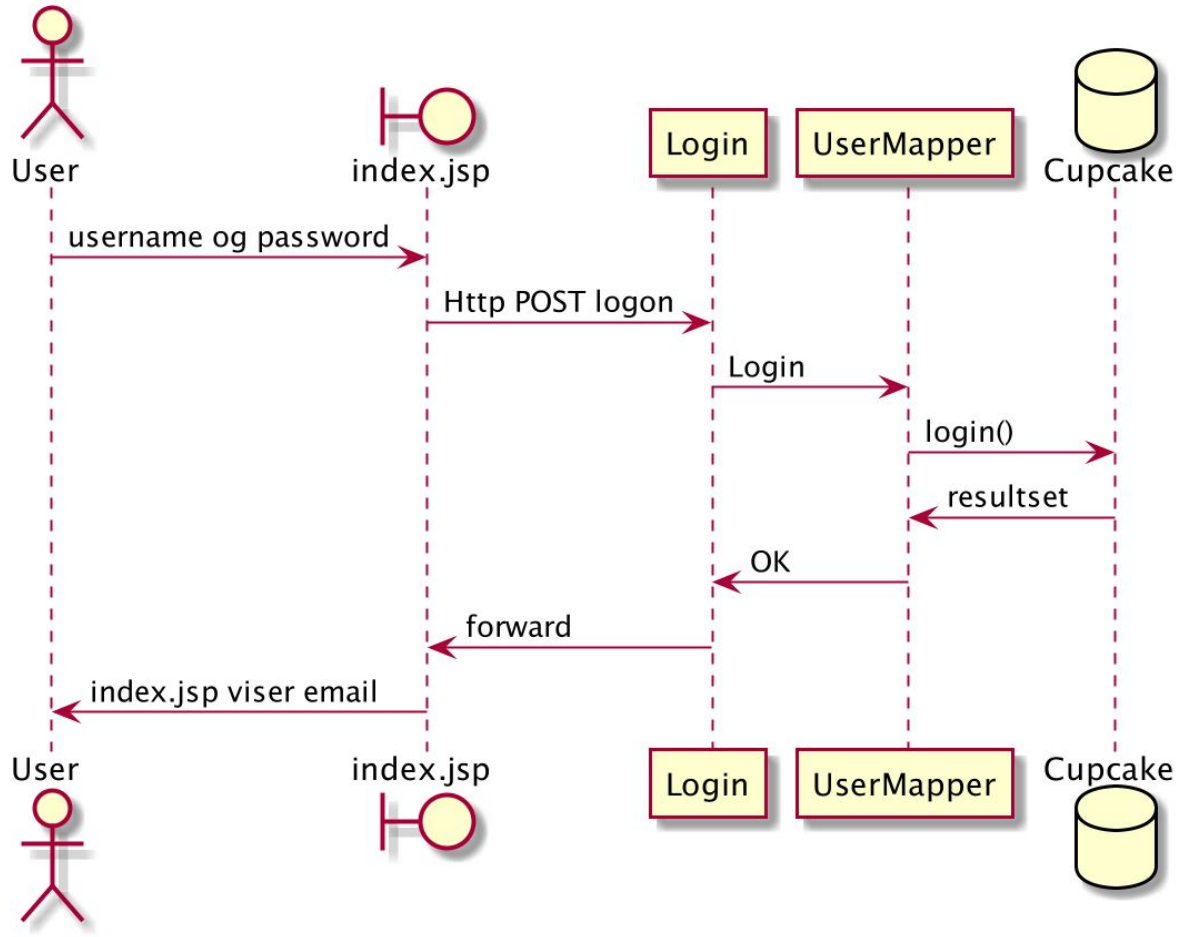
Test

Vi har ikke lavet andre test end ved at teste funktionerne på vores hjemmeside. Her har vi kunne tjekke at ting virkede men vi er ikke i stand til at teste hver enkelt klasse. Dette har vi gjort da vi ikke har været sikre på hvordan eller hvad vi skulle lave junit test på så det gav mening.

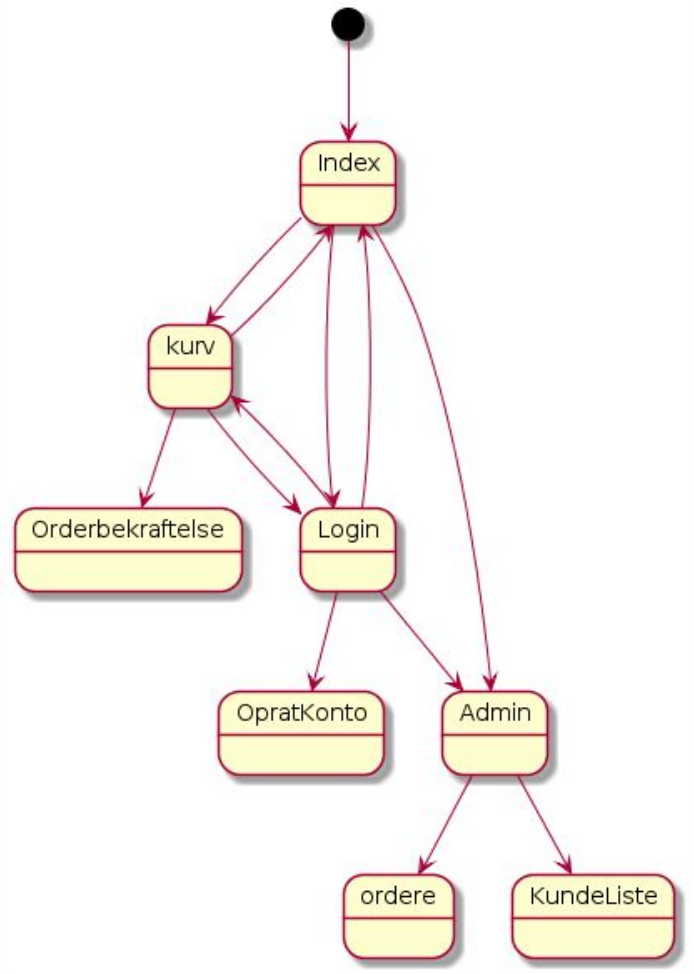
Bilag



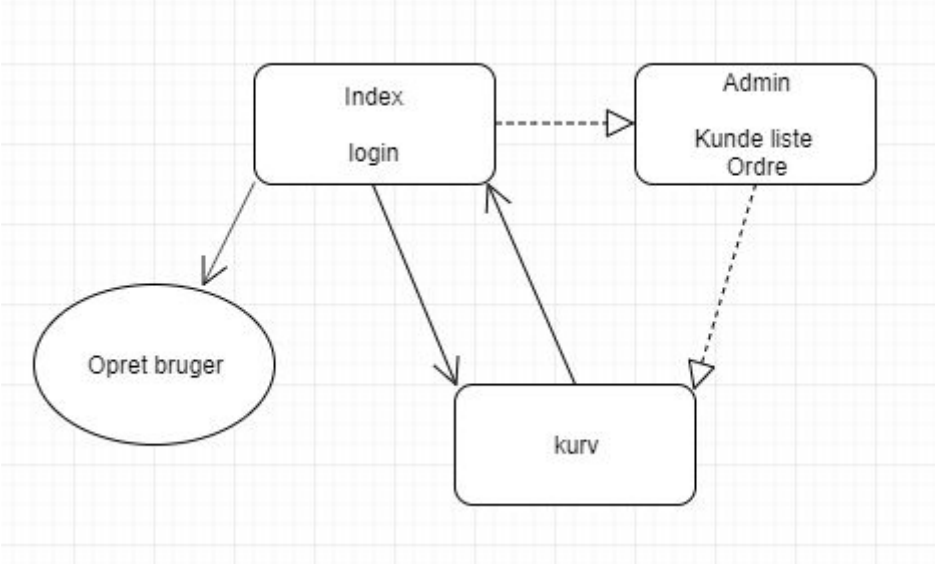
Sekvensdigragram - bestilling og køb



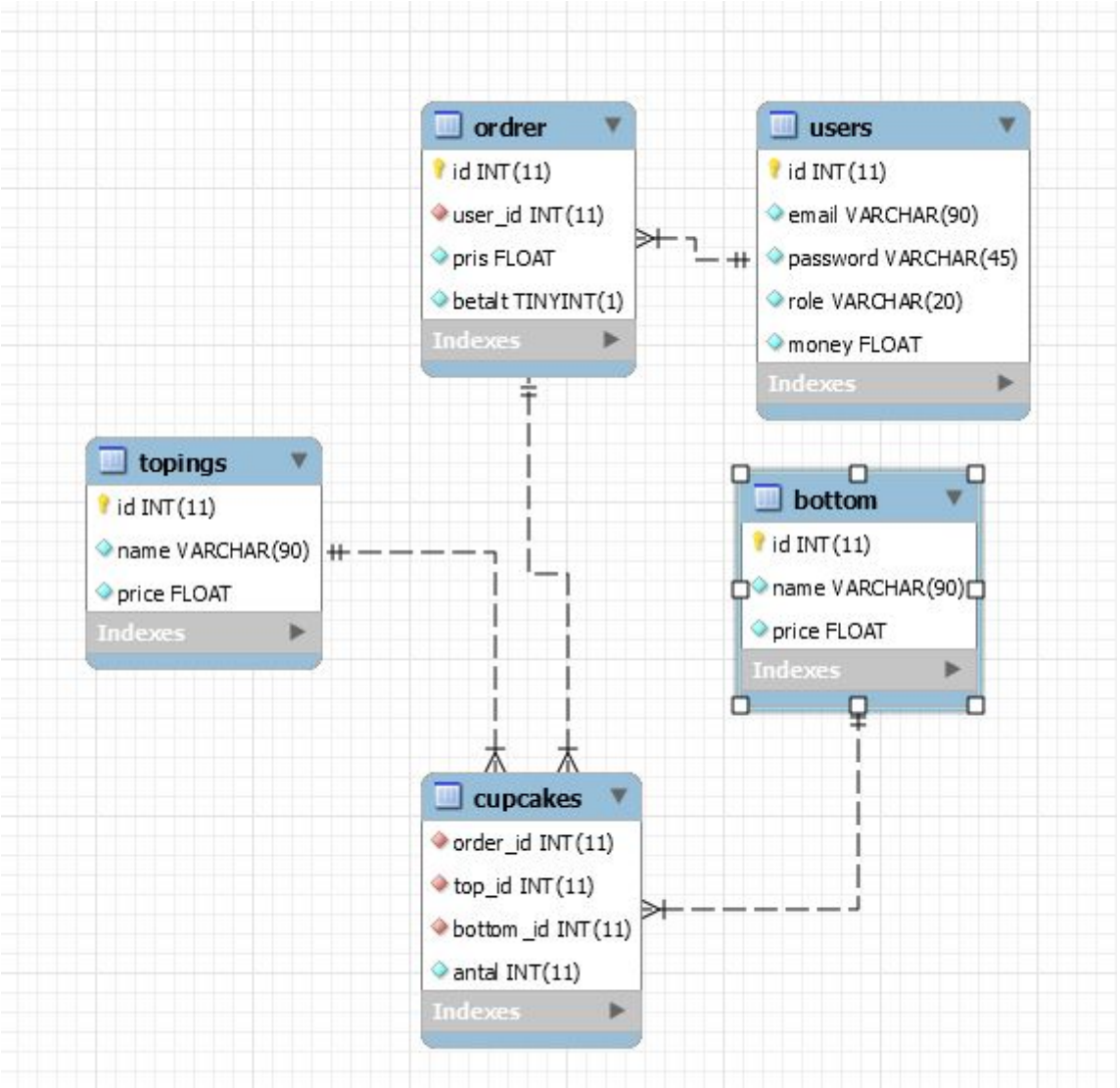
Sekvensdiagram - Login



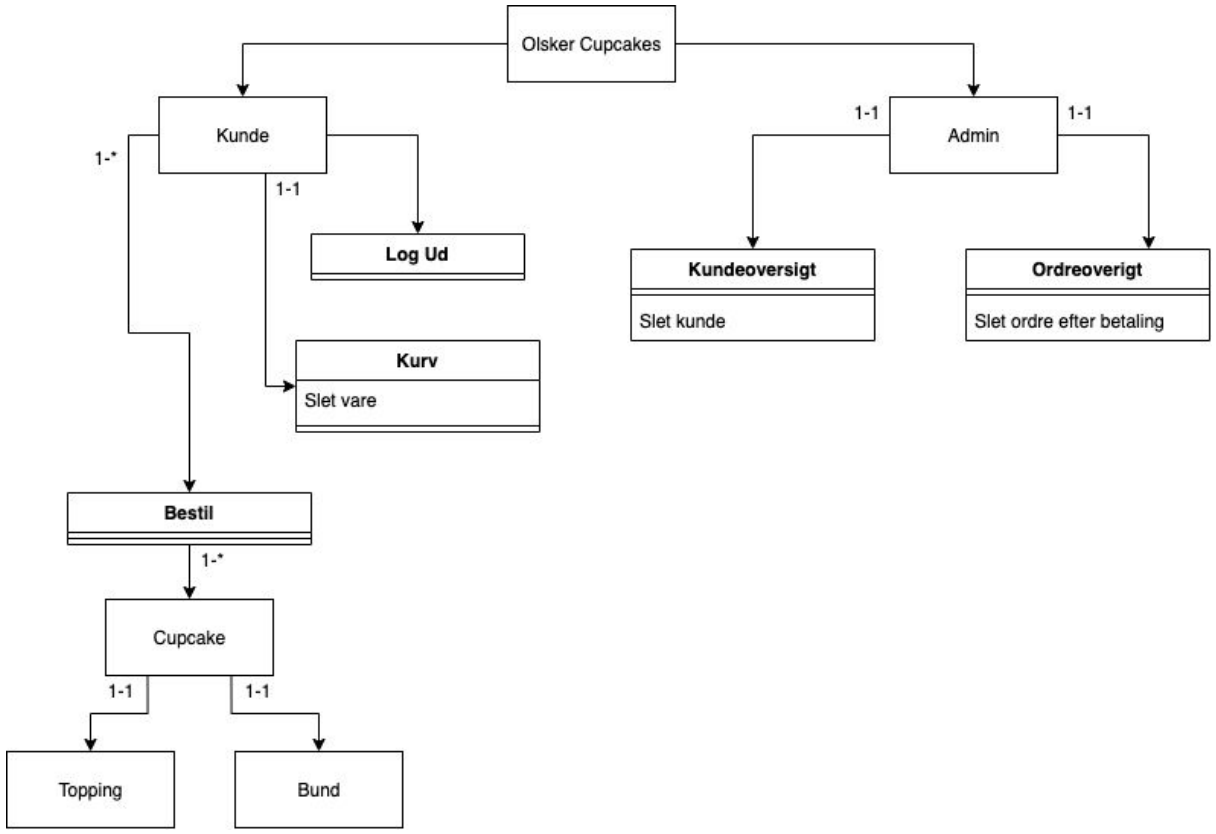
Navigationsdiagram 1-Fra før vi startede på projektet.



Navigationsdiagram 2-Fra efter projektet


ordrer
id INT(11)
user_id INT(11)
pris FLOAT
betalt TINYINT(1)
Indexes
users
id INT(11)
email VARCHAR(90)
password VARCHAR(45)
role VARCHAR(20)
money FLOAT
Indexes
topings
id INT(11)
name VARCHAR(90)
price FLOAT
Indexes
bottom
id INT(11)
name VARCHAR(90)
price FLOAT
Indexes
cupcakes
order_id INT(11)
top_id INT(11)
bottom _id INT(11)
antal INT(11)
Indexes

ER-Diagram



Domæne model