Olsker-cupcakes

29/3 - 2020

Gruppemedlemmer
2. semester D-klassen

Andreas Noer Sejrskild
cph-as510@cphbusiness.dk
Github: @andreasnoer

Emilie Lenschau Hansen
cph-eh141@cphbusiness.dk
Github: @emilielhansen

Mia de Fries
cph-md291@cphbusiness.dk
Github: @miasimone

Rasmus Hansen
cph-rh179@cphbusiness.dk
Github: @rasmusdh

Github
https://github.com/RasmusDH/Olsker-cupcake

Mockups:
https://xd.adobe.com/view/149cc3da-3999-4413-79dc-f5c1fbaea679-62

Andreas Noer, Emilie Hansen, Mia de Fries & Rasmus Hansen
D-klasse

Indledning

Olsker-cupcakes er bornholmsk iværksætter butik som sælger økologiske cupcakes. Butikken har et ønske om en online-shop som deres kunder kan anvende til at bestille cupcakes og derefter afhente i deres butik. Som udgangspunkt kunne de godt tænke sig en webshop hvor at man som kunde kan oprette sig og logge ind for at bestille og betale for cupcakes med valgfri bund og top. Kunden skal kunne se bestillingen og total prisen i en kurv og kunne slette fra kurven. Som kunde skal man kunne se sin email i toppen af siden når man er logget ind. Administratoren skal også kunne logge ind på siden, hvor de skal kunne se alle ordrerne og registrerede kunder. Her skal de også kunne se de bestemte kunders ordrer og kunne fjerne ordrer som er ugyldige.

Krav

Olsker cupcakes ønsker en nemmere måde at tage imod bestillingerne fra kunderne på for på denne måde at kunne udvide deres brand. Fremfor at kunderne skal komme ned i butikken og bestille eller ringe ind kan de på hjemmesiden bestille og betale med det samme og afhente i butikken når bestillingen er klar. Derudover ønsker Olskers Cupcakes en måde at kunne holde styr på alle deres bestillinger og hvem der er kunde hos dem.

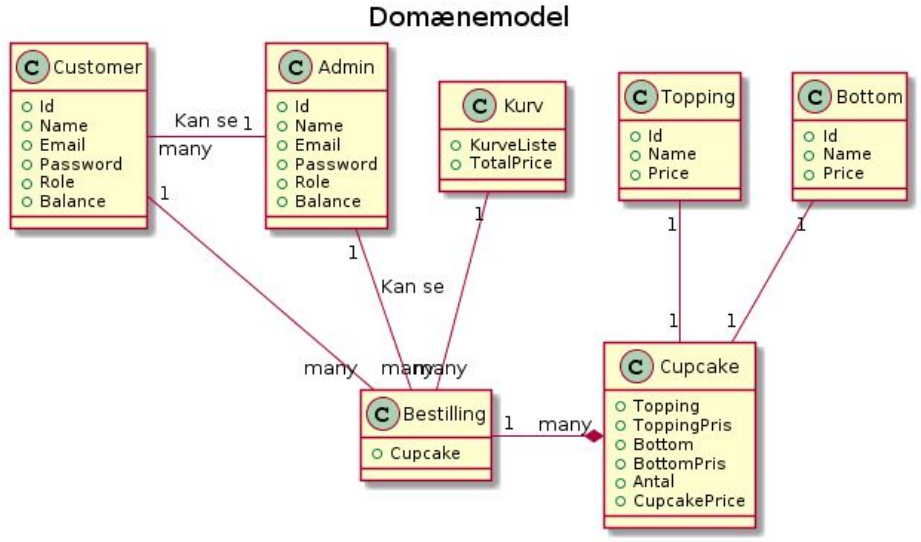
Udover de krav som kunden har sat til os, har vi også selv opsat nogle krav til vores version af hjemmesiden. Den skal være simpel og nem at finde rundt i, sådan så alle kan finde ud af at betjene siden. Derudover har vi et ønske om at hjemmesiden skal have et gennemgående tema.

Til opsætningen af Olskers Cupcakes hjemmeside, har vi benyttet programmet intelliJ IDEA 2019.3. Her har vi alle klasserne og metoderne der for programmet til at køre. Vi har dertil også benyttet MySQL Workbench 8.0 til at oprette vores database. Denne indeholder alle toppings, bunde, kunder og andet. Via intelliJ henter vi data fra databasen.

Hjemmesiden skal åbnes i en webbrowser, derfor har vi benyttet en server via Apache Tomcat 8.5.51. Programmet skal senere lægges op på en online server så det kan køre hele tiden. Dette skal gøres via Digital Oceans droplet.

Andreas Noer, Emilie Hansen, Mia de Fries & Rasmus Hansen
D-klasse

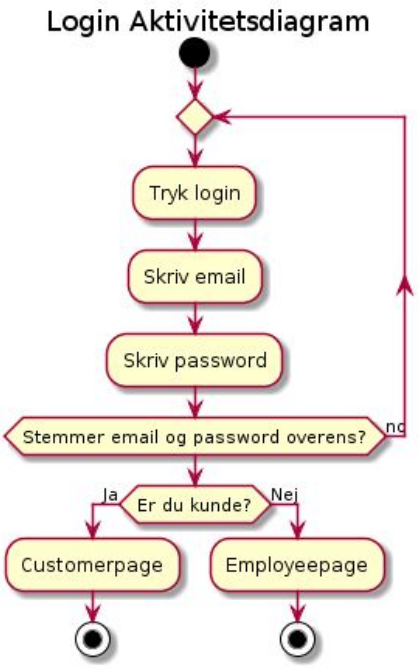
Domænemodel/Klassediagram



I vores domænemodel har vi 7 klasser, som alle sammen kører ud fra bestillinger. Fra bestillinger kan man designe sine cupcakes: vælge en top og en bund, hvor de så bliver lavet til en bestilling. Man kan så trykke tilføj til kurv, således bliver bestillingen tilføjet til den kurveliste der bliver oprettet når man logger ind.

Administratoren administrerer bestillingerne og kunderne, derfor er administratoren ikke knyttet til andet end dem.

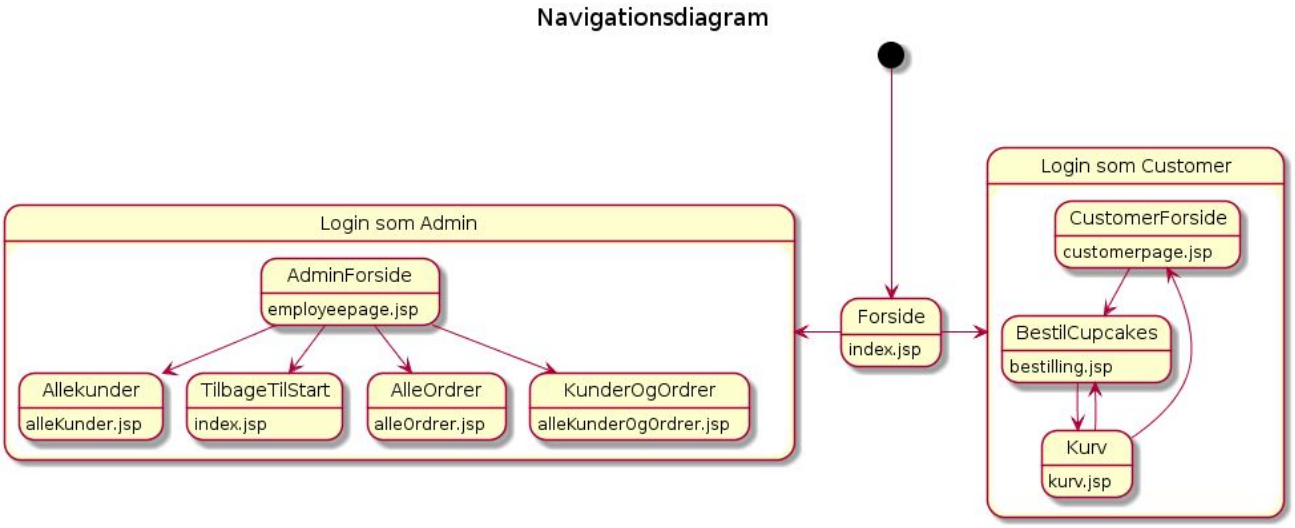
Aktivitetsdiagrammer



Bestillings-aktivitetsdiagrammet beskriver alt hvad kunden skal gøre for at bestille cupcakes. Det skal lige bemærkes at vi antager at kunden allerede er logget ind når der trykkes på bestillings-knappen. Ellers ville man blive sendt til login siden, da man skal være logget ind for at kunne bestille.

Login-aktivitetsdiagrammet beskriver alt hvad kunden skal gøre for at logge ind. Hvis emailen og passwordet ikke stemmer overens, kommer man til index-siden, og hvis de gør, kommer man til den side, hvis rolle passer til: Hvis man er kunde kommer man til kundesiden og hvis man er employee kommer man til employee siden.

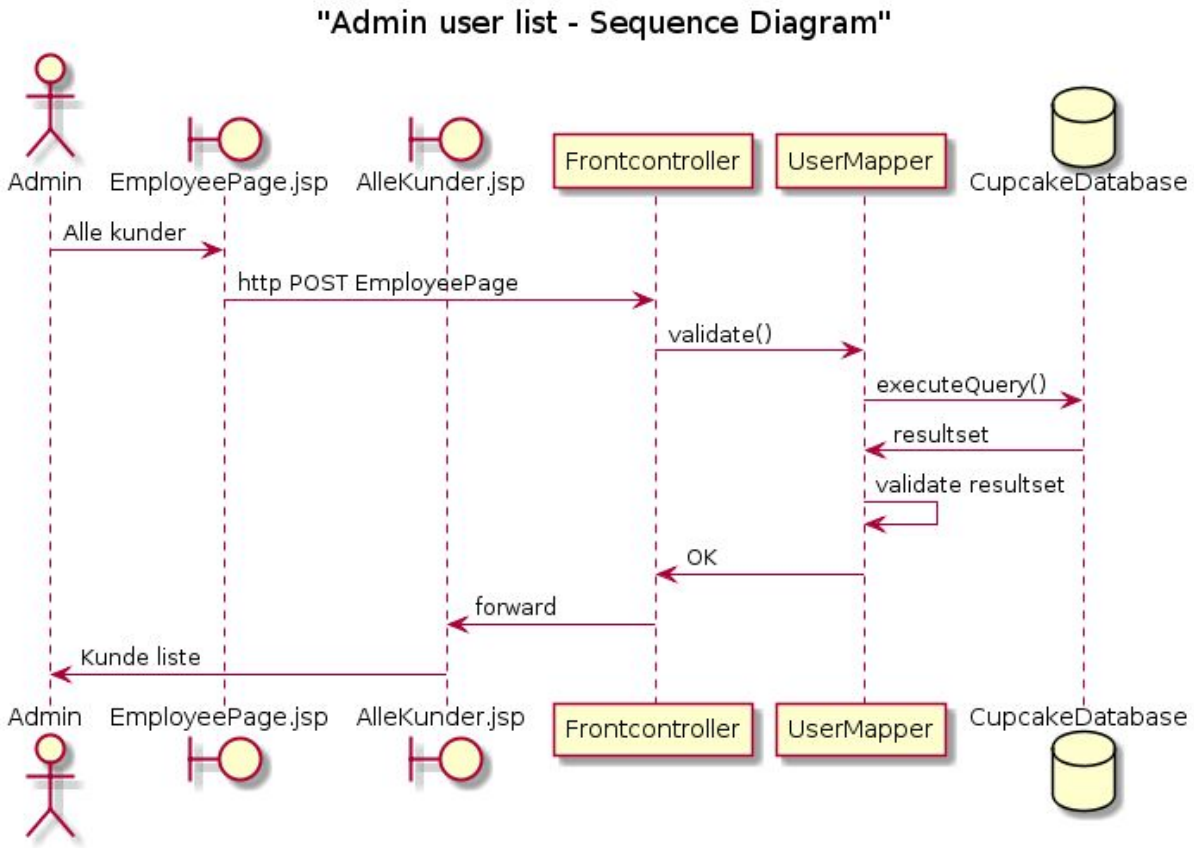
T*ilstandsdiagram*/Navigationsdiagram



Man starter på index-siden/forsiden hvor man herfra kan tilgå login-siden. Her kan man logge ind som enten administrator eller kunde. Logger man ind som kunde kommer man tilbage til forsiden hvor man herefter kan komme ind på bestillingssiden. Forsøger man at komme ind på bestillingssiden uden at være logget ind, bliver man sendt direkte til login siden. Efter man har bestilt cupcakes, bliver man sendt til kurven. Fra kurven kan man gå tilbage til forsiden eller tilbage til bestillingssiden. Der er en navigationsbar i toppen af hver side, her kan man også tilgå kurven og login-siden. Logger man ind som administrator, kommer man til employee-siden. Denne side kan kun blive tilgået hvis man er logget ind med et administratorlogin. På administrator siden kan man indsætte beløb på brugernes konto. Herfra kan man tilgå en anden side hvor man kan se alle kunderne, en side hvor man kan se alle ordrerne og en side hvor man kan se ordrerne og de kunder de hører til.

Andreas Noer, Emilie Hansen, Mia de Fries & Rasmus Hansen
D-klasse

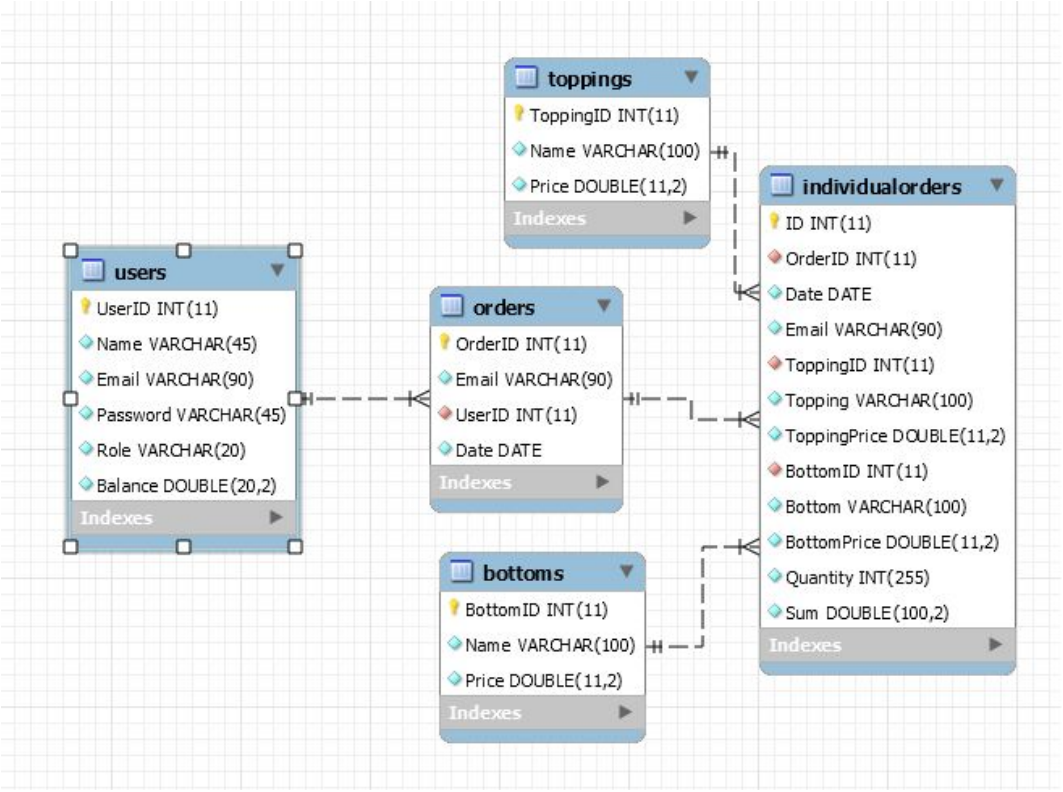
Sekvensdiagram



Vores sekvensdiagram beskriver processen i at gå ind og se en kundeordre og, hvad administratoren af siden skal kunne gøre. På admin siden kan administratoren trykke på kunde knappen, hvorefter der bliver sendt en http POST til Frontcontroller som validerer dette. Derefter bliver det sendt til UserMapper, der henter informationen fra databasen og lægger det ind i en ArrayList. Denne ArrayList bliver så valideret og sendt tilbage til Frontcontroller, hvor det bliver forwarded tilbage til AlleKunder.jsp siden der vises til administratoren.

Andreas Noer, Emilie Hansen, Mia de Fries & Rasmus Hansen
D-klasse

ER-diagram



Hvert table har en primary key (de gule), f.eks. så har "bottoms" BottomID.

For at få de forskellige tables til at hænge sammen så har "individualorder" og "orders" foreign keys, f.eks. CustomerID og OrderID (de røde).

Ved brug af foreign keys i "individuelorders", kan vi hente topping navnet og topping prisen ind ved blot at forbinde "toppings"-table med "individuelorders"-table på ToppingID.

I vores program henter vi toppings og bottoms fra databasen og udskriver det i dropdown menuer således at en kunde kan vælge mellem de forskellige toppings og bunde. Når ordren er valgt og betalt, bliver den lagt ind i både "orders" og "individualorders". Fra employee-siden bliver lister af customers, orders og individualorders hentet ind fra databasen og udskrevet på skærmen. Således kan administratoren se alle kunderne, ordrerne og hver kundes ordre, hvilket opfylder 2 af user-storiesne.

Administratoren skal også kunne slette ordre hvilket gøres via ordreID'et i programmet. Den sletter altså den ordre der har det OrdreID som indtastes.

Hele opsætningen af databasen der er connected til java, går ud fra CRUD.

CREATE: Man kan oprette sig som kunde

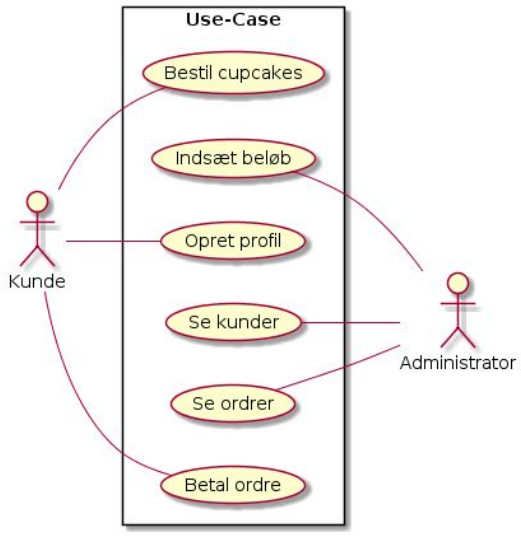
READ: Listen af kunder bliver printet ud på skærmen

UPDATE: Man kan opdatere kundens saldo

DELETE: Man kan slette en ordre

Andreas Noer, Emilie Hansen, Mia de Fries & Rasmus Hansen
D-klasse

Use-Case



Vores use-case diagram beskriver hvilke krav der er til henholdsvis hvad kunden skal kunne, og hvad administratoren af siden skal kunne. Som udgangspunkt er vi gået efter de seks første user-stories. Kunden skal kunne bestille cupcakes, oprette en profil og betale den ordrer som de har lavet. Administratoren skal kunne indsætte beløb på kundens profil, se hvilke kunder som har en profil på siden, og se alle ordrerne og hvilke ordrer der hører til hvilken kunde.

Implementeret vs planlagte dele af systemet

Vi har opnået de fleste af vores og kundens mål med hjemmesiden. Vi har fået udviklet et program hvor en kunde kan bestille cupcakes og betale, dog følger databasen ikke trop, det vil sige at ordren ikke bliver lagt ind i databasen som egentlig var tanken. Dog bliver listen af cupcakes clearet og den totale pris bliver nulstillet.
Kunden kan ligeledes oprette en profil, og dette er nødvendigt for at kunne bestille og betale for cupcakesne. Dette gøres via login siden hvor der er implementeret en registrer knap. Når man har registreret sig bliver man sendt til forsiden hvor man kan bestille cupcakes.
Når man er logget ind bliver ens email vist på alle sider i navigationsbaren, dette gælder også for adminen.
Adminen kan indsætte et beløb på en kundes konto og kan ligeledes se ordrerne der er bestilt, kunderne samt alle kunderne og deres ordre. De bliver hentet fra databasen og udskrevet hvis man trykker på en knap.

De ting som vi ikke har fået implementeret men som vi kunne tænke os at få lavet hvis vi havde mere tid ville være at få de funktioner der endnu ikke fungere optimalt til at virke.
Vi vil også lave to yderlige sider på vores hjemmeside. En side hvor man kan se de ingredienser som er blevet gjort brug af, da Olsker cupcakes er en økologisk butik og går op i deres ingredienser, samt en side omkring Olsker Cupcakes, hvem de er og deres kontaktinformation.

Vi har i sidste øjeblik fundet en fejl som er at listen på ny.jsp ikke vil opdatere. Derudover er der opstået en ny fejl som gør, at vores ordreliste ikke fremvises, dette arbejder vi på at få fikset over de næste par dage.

Derudover mangler vi også at få lavet sådan så man kan opdatere listerne, men hvis vi havde mere tid ville vi gøre sådan at de kan opdatere korrekt. Dette ville vi løse ved at oprette nye jsp sider til hver

knap, benytte os af Inizialiseren og kalde "hente" metoden i Mapperne for at få udskrevet listerne af ordre, kunder og kundernes ordre.

Bruger input validering
Brugerne skal logge ind for at kunne bestille cupcakes. Dermed bliver deres mail samt password valideret ud fra det tilsvarende i databasen for at kunne bestille. Hvis man indtaster en email og et password der ikke stemmer overens i forhold til databasen, så kommer man ikke ind på bestillingssiden. Ligeledes skal man vælge et antal cupcakes for at kunne bestilles, ellers vil programmet melde at der er en fejl. På den måde kan man ikke bestille "tomme" cupcakes.

https://datsoftlyngby.github.io/dat2sem2020Spring/uge13/javadoc.html

Særlige forhold

Bruger input validering

Brugerne skal logge ind for at kunne bestille cupcakes. Dermed bliver deres mail samt password valideret ud fra det tilsvarende i databasen for at kunne bestille. Hvis man indtaster en email og et password der ikke stemmer overens i forhold til databasen, så kommer man ikke ind på bestillingssiden. Ligeledes skal man vælge et antal cupcakes for at kunne bestilles, ellers vil programmet melde at der er en fejl. På den måde kan man ikke bestille "tomme" cupcakes.

Brugertyper

Vi har lavet to slags brugertyper: en kunde og en administrator. Når man logger ind kommer man som bekendt til enten customerpage eller employeepage. Dette validerer den ud fra Role i users table.

Session

Vi benytter os ofte af at gemme attributter i session. Bl.a. til brugeren: når en bruger logger ind, sætter vi dens f.eks. email (linje 33 i nedenstående), som vi udskriver i navigationsbaren når brugeren er logget ind. Her kunne vi også udskrive en kundes balance (bliver sat på linje 32 i nedenstående). Her er et snapshot af vores login klasse der gør netop det:

```
public class Login extends Command {

    @Override
    String execute( HttpServletRequest request, HttpServletResponse response ) throws LoginSampleException {
        String email = request.getParameter( s: "email" );
        String password = request.getParameter( s: "password" );
        User user = LogicFacade.login( email, password );

        HttpSession session = request.getSession();
        Kurv kurv = (Kurv) session.getAttribute( s: "kurv");
        if (kurv == null) {
            kurv = new Kurv();
        }

        session.setAttribute( s: "kurv", kurv);
        session.setAttribute( s: "user", user );
        session.setAttribute( s: "navn", user.getName());
        session.setAttribute( s: "role", user.getRole() );
        session.setAttribute( s: "balance", user.getBalance());
        session.setAttribute( s: "email", email);  // ellers skal man skrive  user.email på jsp siderne og det

        return user.getRole() + "page";
    }
```

Andreas Noer, Emilie Hansen, Mia de Fries & Rasmus Hansen
D-klasse

Særligt:

En bestemt ting vi har gjort er, at kurv klassen bliver sat som en session attribute når brugeren logger på, så på denne måde køre den fra bruger til bruger uafhængig af hinanden.

Vi har også lavet en exception håndtering ved login, hvor den melder tilbage med en error boks, hvis du har indtastet en forkert kombination som ikke kan findes i databasen.