Oleks Cupcakes

2. Semester midvejs projekt

Marts 2020

ISSUE:
Gruppe nummer: 123

Medlemmer (Skriv jeres navn og github acc navn)
Lars Poulsen -Lars262
Jonathan Zielinski -j-zie
Benjamin Iglesias -BenjaminIglesias
Patrick Jahn -PatrickJahn
Mads Johan Bech -1337k1ng

Indholdsfortegnelse

Introduktion

Oleks cupcakes er en web applikation skrevet i Java 8.0 og Oracle's web løsning, JSTL, som understøtter kerne operationerne fælles for de fleste web teknologier. Navnligt evnen til at modtage http/get og sende http/posts. Vores applikation læser, skriver, opdatere og sletter indhold, og kræver derfor persistent data, til dette formål bruger vi MYSQL, som er hosted på en ubuntu server. Bindeleddet mellem klienten og vores web application er tomcat, en http server, som selvfølgelig også er hosted på vores ubuntu server. Kildekoden er centraliseret omkring et design mønster, *command mønstret*, hvilket betyder en abstrakt klasse har kendskab til de metoder vi gerne vil sende afsted, og ved ved hvornår de skal eksekveres.

Baggrund

Virksomheden Olsker Cupcakes®, som driver en stor Cupcake butik i Olsker på Bornholm, har bedt om et online system, der gør det muligt for deres kunder at bestille cupcakes på forhånd. Til dette system har de angivet nogle krav og ønsker:

Deres kunder skal kunne oprette en konto i systemet, for således at kunne bestille og betale cupcakes med en valgfri bund og top, sådan at de senere kan komme forbi butikken i Olsker og hente deres ordre. Kontoen skal kunne oprettes ved hjælp af deres e-mail og et password, som de efter oprettelsen kan bruge til at logge ind på systemet igen.
Endvidere skal de kunne se deres valgte ordrer i en "indkøbskurv" , samt den totale pris, og eventuelt fjerne en ordre, inden de bestiller.

Virksomheden selv skal som administrator kunne logge på systemet via en email og et password, hvor de kan indsætte beløb på kundernes konto således at kunderne kan

betale. De skal desuden også kunne se alle kunderne der er oprettet i systemet og deres ordrer. Derudover skal de have mulighed for at fjerne en ordre.

Teknologi valg

Server:

- Virtuel Ubuntu 19.10 x64

Database:

- MySQL

IDE:

- IntelliJ 2019.3.3 / 2019.3.4

Sprog:

- Java 8.0

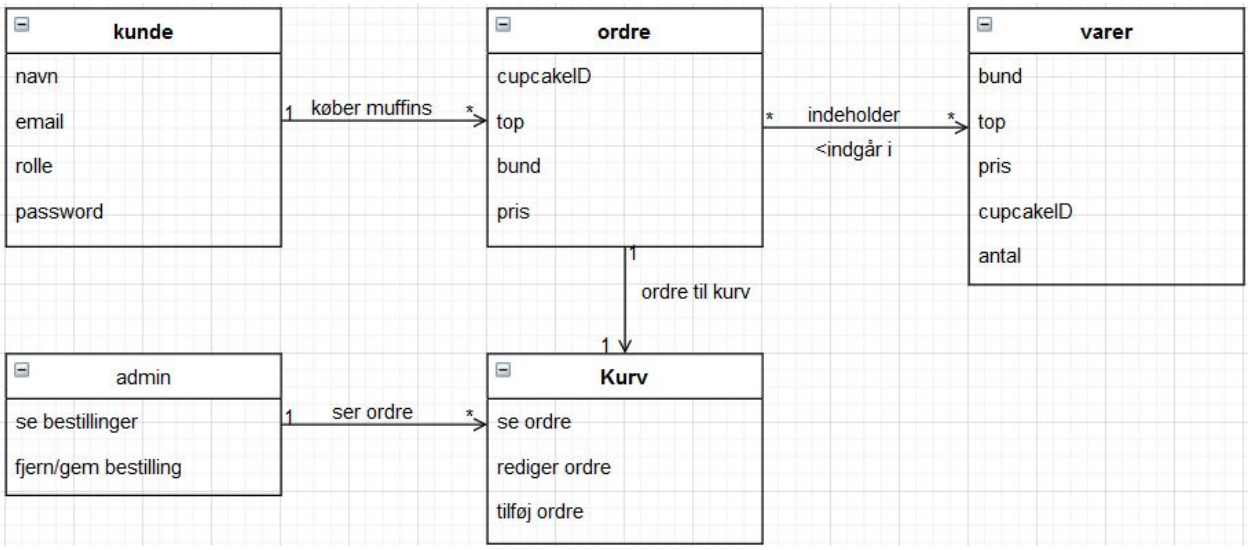
Build:

- Maven

API:

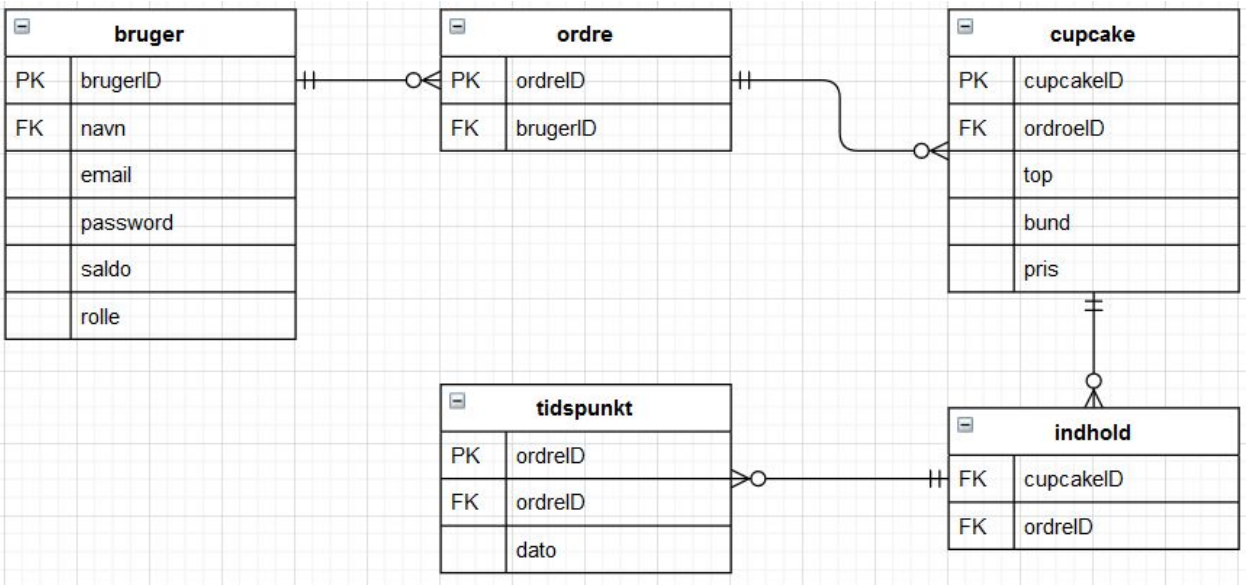
- Apache Tomcat 8.5.51
- JDBC

Domæne model



På diagrammet ser vi relationerne mellem klasserne samt de enkelte klassers attributter. Som en lille gennemgang har vi først en kunde som har attributterne: navn, email, rolle og password. Kunden køber muffins, dette tager dem over i ordre, hvor en kunde kan have flere ordre, men hver ordre kan kun være tilknyttet en kunde. altså er det en 1-* relation. Det bringer os videre til ordre som har attributterne: cupcakeID, top, bund og pris. Ordre tager udgangspunkt i varer som har de samme attributter samt antal. forholdet er en *-* relation da en ordre kan indeholde mange varer og varer indgår i mange ordre. Herfra bevæger vi os tilbage til ordre da den også her en relation til kurv. Hver ordre kan kun have en kurv og hver kurv kan kun være tilknyttet en ordre. Så dette er en 1-1 relation. Kurven har attributterne: se ordre, rediger ordre og tilføj ordre. Så kommer vi til admin som har attributterne: se bestillinger og fjern/gem bestilling. Forholdet er 1-* da admin kan se mange kurve, men hver kurv kan kun have en admin. (Det er dog muligt at have flere admin og der ved vil det være en *-* relation).

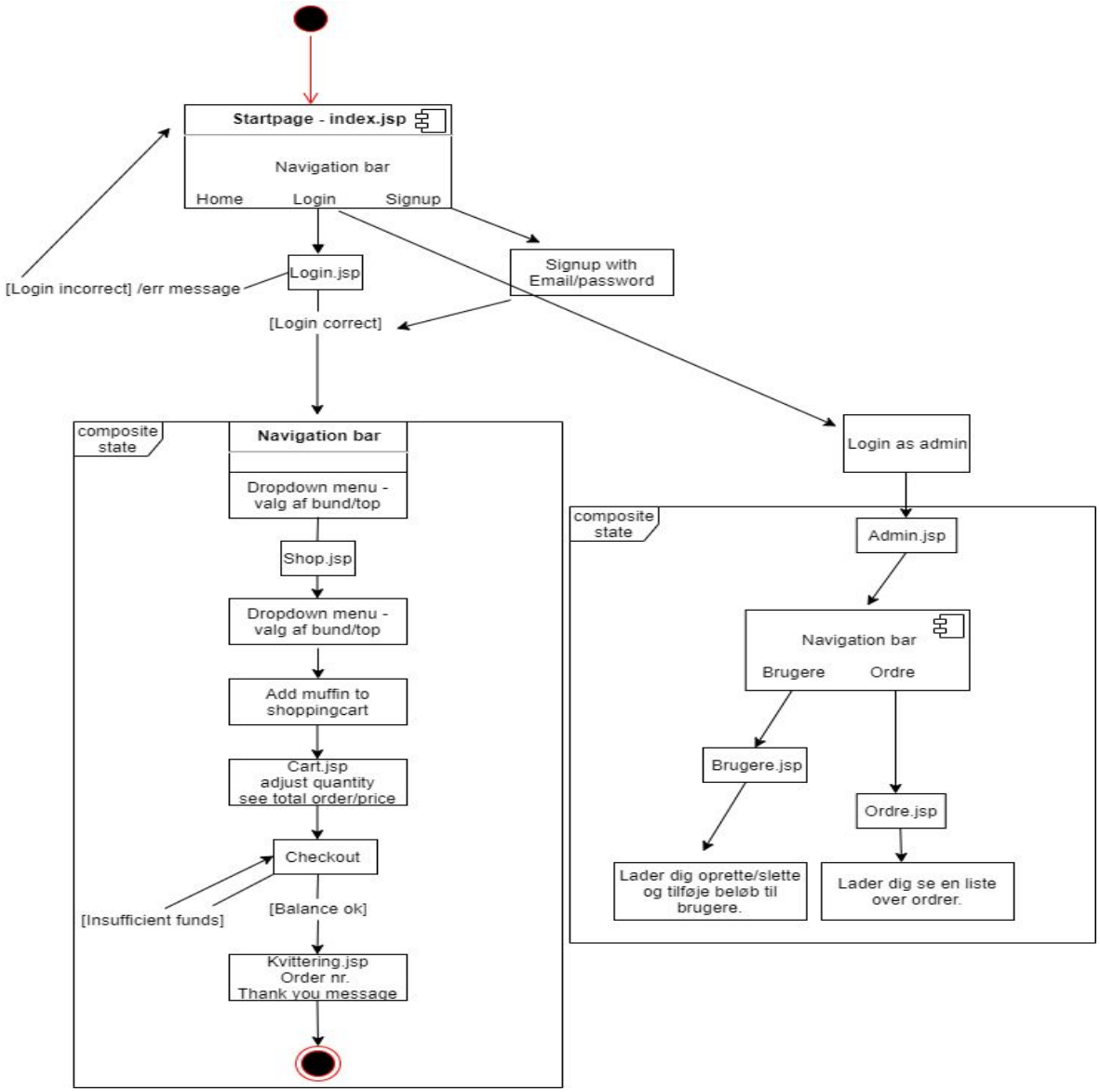
ER diagram



Følger vi diagrammet har vi først en tabel der hedder bruger. Den har attributterne: brugerID som er en primary key, navn som er foreign key, email, password, saldo, rolle. Fra bruger går vi videre til tabellen cupcake som har attributterne: cupcakeID som er primary key, top, bund og pris. Så kommer vi til ordre tabellen som har attributterne: ordreID som primary key og bruger ID som foreign key. Derefter går vi videre til indholds tabellen som har attributterne: cupcakeID som foreign key og ordreID some Foreign key. Til sidst har vi tidspunkts tabellen som har attributterne: ordreID som primary key, ordreID some foreign key og dato.

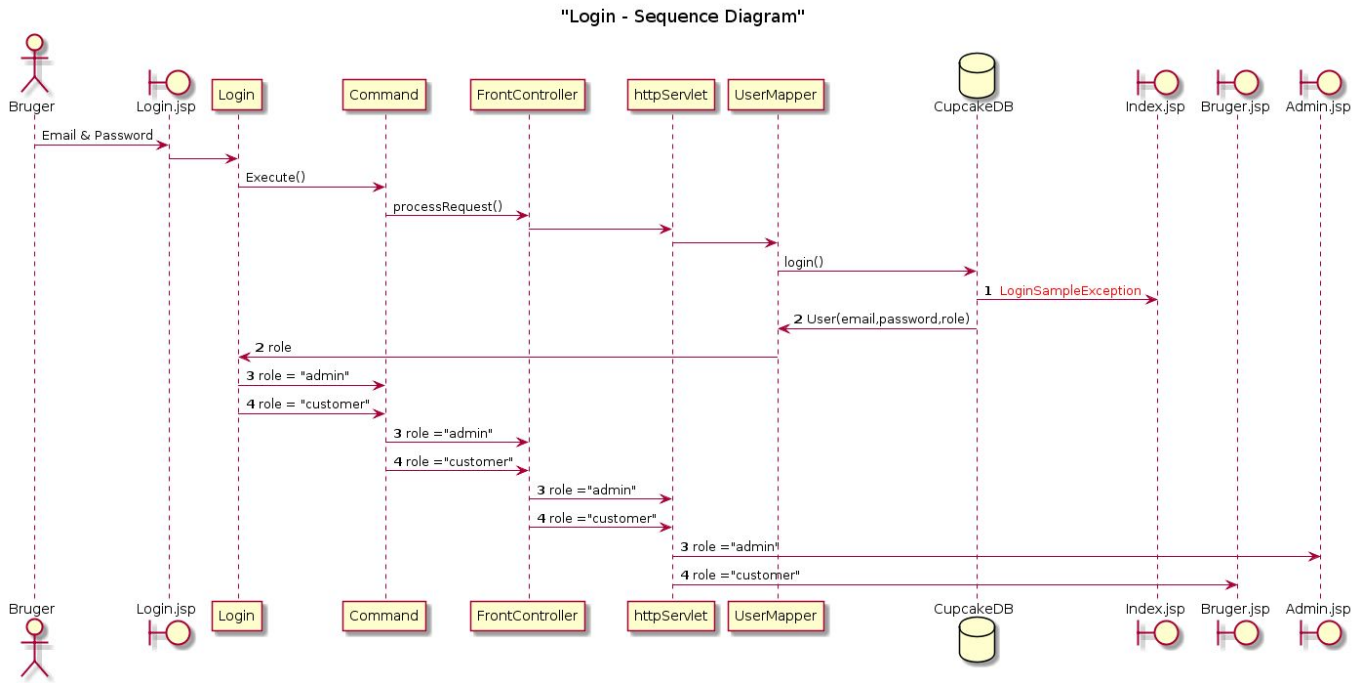
Navigationsdiagram

En fælles navigations-bar er benyttet på tværs af alle sider.
Ved login er der en ny side "shop" som bliver tilgængelig.
Når en med administrator privilegier logger ind, vil navigations-baren have rettigheder, til at kunne rette/slette ordrer, og tilføre likviditet til kontoer.



Sekvens diagrammer

Et sekvensdiagram der i dybden viser loginprocessens trin, både i tilfælde af succes og fejl. I et succesfuldt tilfælde vil destination som vist være afhængig af brugerens rolle.



Særlige forhold

Sessionen gemmer oplysninger tilknyttet til bruger. Hvor af disse består af navn, email, saldo, rolle. Derudover har vi også bruger listen som gemmes i sessionen samt ordrelisten. Cupcakes bliver ikke gemt i session scope, men bare lagt i en array liste i en klasse kaldet muffin kurv. Vi burde måske gøre det anderledes. Dette gør eksempelvis at kurven ikke bliver tømt når man kalder session.invalidate. Vi har valgt at lave sikkerhed ved at kræve brugeren logger på med et password som bliver tjekket op mod databasen.

I/O operationer, input som oftest er fra databasen, er skrevet i try/catch blokke, og deres potentielle fejl situationer bliver håndteret således.
Brugerinput bliver valideret ved if statement hvor programmet kigger efter den rigtige syntax i e mail for eks. Hvis der mangler et @.

Vi har valgt to brugertyper, en kunde/bruger og en admin. Dette bliver gjort ved at tildele dem en rolle inde i programmet.

Status på implementation

Vi mangler noget css til ordre liste da denne kan forekomme lettere rodet. Muligvis lave en bedre håndtering af password i forhold til bedre sikkerhed. Mangel på at vælge antal af muffins i shoppen. Login throws exception som kaster brugeren ud hvis de ikke skriver rigtigt i stedet for at prompte til at skrive password igen.