Cupcake

Deltager

Navn: Jakob Staudal

Email: cph-js554@cphbusiness.dk

GitHub-brugernavn: staudal

Projekt

Dato: 17. nov 2022

Gruppe: *12*

Link til demo: YouTube

Indholdsfortegnelse

Indledning

Denne rapport er baseret på en hjemmeside, som er blevet udviklet for at hjælpe en cupcake-virksomhed (Olsker cupcakes) med at sælge en masse cupcakes. Førhen solgte Olsker sine cupcakes i deres fysiske butik, men vil gerne med på den teknologiske bølge og sælge en masse cupcakes. På hjemmesiden skulle man være i stand til at oprette sig som bruger, logge ind, bestille, se en indkøbskurv, se sine ordre og senerehen kunne afhente sin ordre i den fysiske butik.

Baggrund

Olsker sælger deres cupcakes fra en fysisk butik, hvor de har problemer med at tiltrække kunder til butikken. De har længe gået med tanken om at investere i en platform, hvor kunder kan lave deres egne cupcakes (vælge kagebund og topping), købe dem og senere afhente dem i den fysiske butik. Senerehen overvejer Olsker også at udvide med levering, så kunderne ikke selv behøver at møde op fysisk i butikken for at afhente kagerne. Men det vil de ikke fokusere på i første omgang.

For at hjælpe Olsker med deres problem har jeg udviklet en hjemmeside, som er kodet i følgende:

- **Backend:** Java (JSP & Servlets) & MySQL
- **Frontend:** HTML, CSS

Jeg har brugt programmeringsværktøjet IntelliJ 2022.2.3 (Ultimate Edition) til at programmere webshoppen.

Teknologivalg

Nedenstående teknologier er blevet brugt for at programmere hjemmesiden:

- **IDE:** IntelliJ 2022.2.3 (Ultimate Edition)
- **Database værktøj:** MySQL Workbench 8.0.30
- **Server:** Apache Tomcat 10.0.26
- **JDK:** OpenJDK 19
- **Backend sprog:** Java
- **Frontend sprog:** HTML, CSS

- **Frontend framework:** Bootstrap 5

Krav til systemet

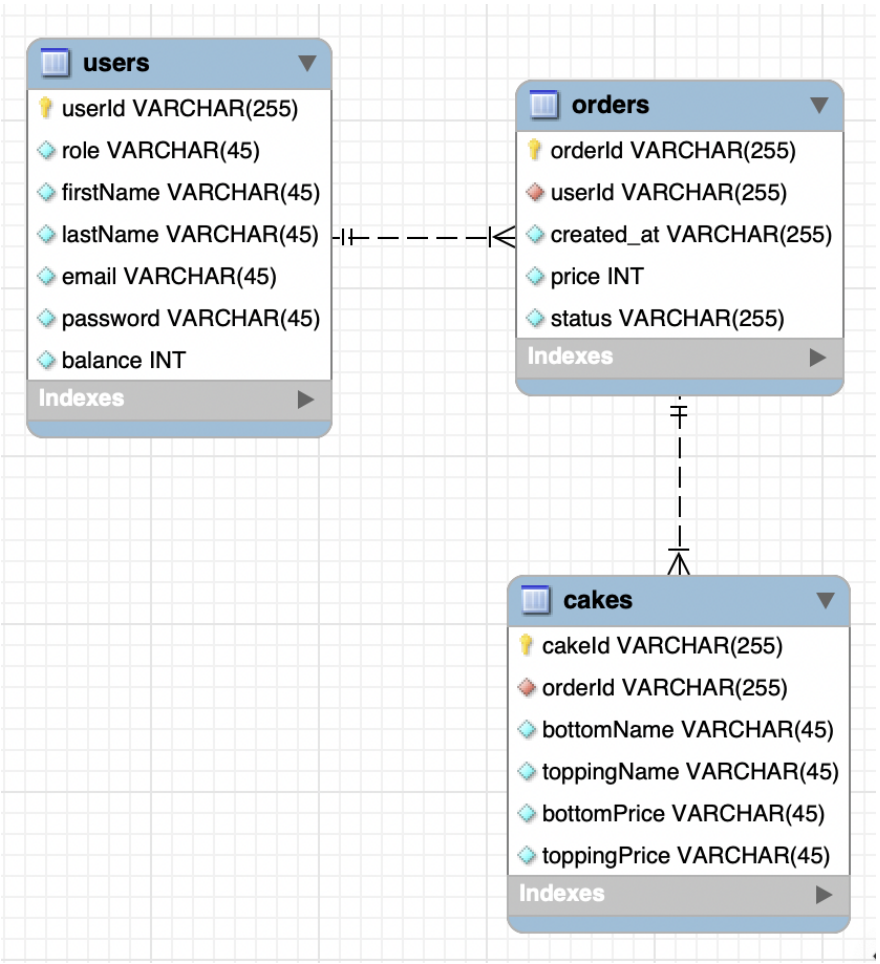
Olsker har i længere tid gået og bakset med at kunne håndtere kunderne, som kommer ind i deres fysiske butik. Køerne bliver meget lange, fordi Olsker kun har 2 ansatte. Èn som tager imod kundernes bestillinger og én som laver cupcakes. Ved at bruge webshoppen kan begge ansatte lave cupcakes uden at skulle tage imod bestillingerne fysisk.

Jeg er blevet givet de følgende user stories:

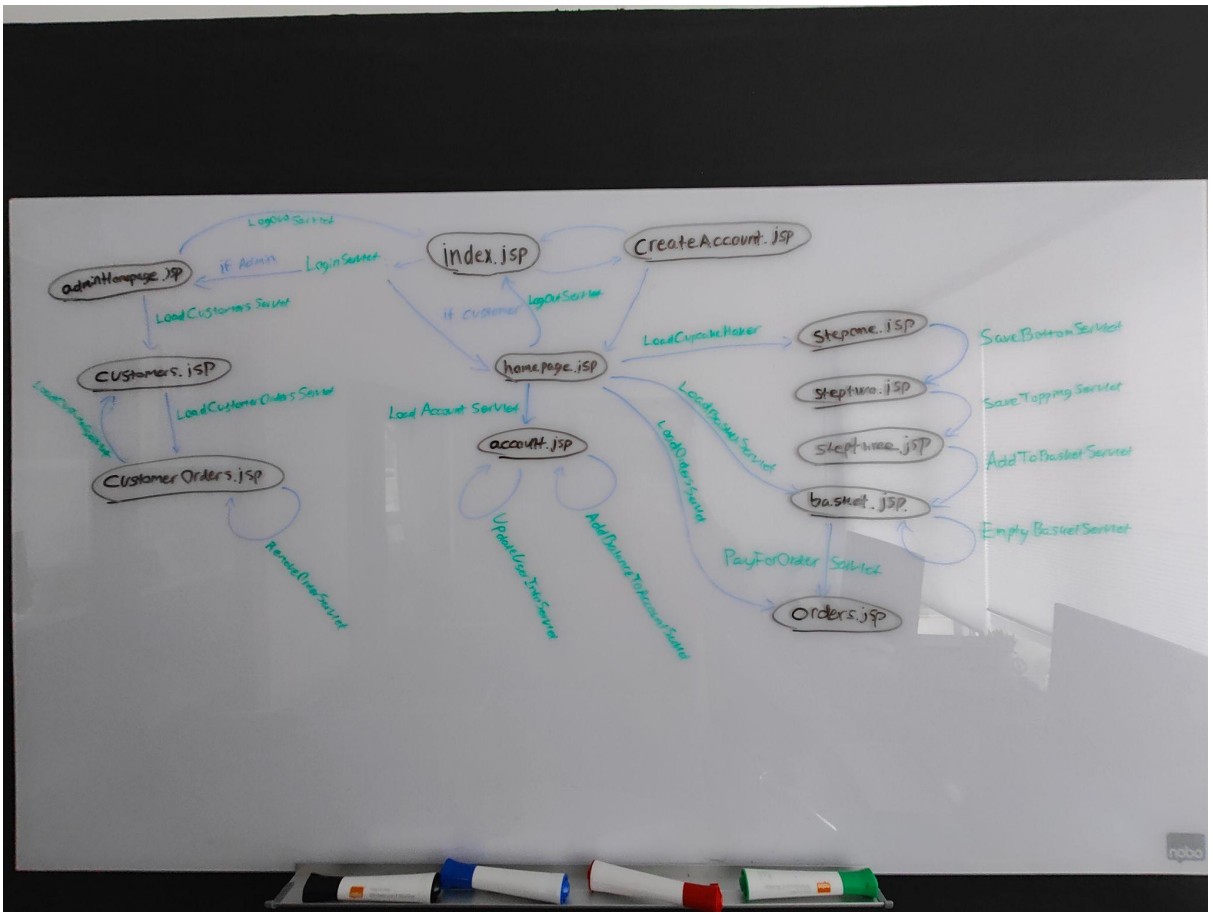
1. Som kunde kan jeg bestille og betale cupcakes med en valgfri bund og top, sådan at jeg senere kan køre forbi butikken i Olsker og hente min ordre.
2. Som kunde kan jeg oprette en konto/profil for at kunne betale og gemme en en ordre.
3. Som administrator kan jeg indsætte beløb på en kundes konto direkte i MySql, så en kunde kan betale for sine ordrer.
4. Som kunde kan jeg se mine valgte ordrelinier i en indkøbskurv, så jeg kan se den samlede pris.
5. Som kunde eller administrator kan jeg logge på systemet med email og kodeord. Når jeg er logget på, skal jeg kunne se min email på hver side (evt. i topmenuen, som vist på mockup’en).
6. Som administrator kan jeg se alle ordrer i systemet, så jeg kan se hvad der er blevet bestilt.
7. Som administrator kan jeg se alle kunder i systemet og deres ordrer, sådan at jeg kan følge op på ordrer og holde styr på mine kunder.
8. Som kunde kan jeg fjerne en ordre fra min indkøbskurv, så jeg kan justere min ordre.
9. Som administrator kan jeg fjerne en ordre, så systemet ikke kommer til at indeholde udgyldige ordrer. F.eks. hvis kunden aldrig har betalt.

Diagrammer

EER diagram



Navigationsdiagram



Særlige forhold

Under udviklingen af webshoppen har jeg gjort mig de følgende overvejelser:

1. Gemt user i session scope

Når en bruger opretter en konto eller logger ind, bliver der oprettet et User objekt, som har en constructor, som tildeler objektet disse variabler: id, email, kodeord, fornavn, efternavn, rolle (admin eller kunde), indkøbskurv (objekt). User-objektet bliver derefter gemt i et session scope, så programmet hele tiden ved, hvilken bruger der køber noget eller vil se sine ordre. Hver gang der bliver foretaget en handling, som omfatter User-objektet, bliver det også synkroniseret til databasen, så user-objektet altid stemmer overens med databasen.

2. Delt valg af cupcake op i 3 dele

For at gøre cupcake oplevelsen så god som muligt, har jeg valgt at dele processen hvor man laver sin cupcake op i 3 steps. Først vælger man en kagebund. Når kunden trykker på den valgte kagebund, bliver der oprettet et nyt kagebund-objekt, som får tildelt et navn og en pris. Det samme sker når brugeren bagefter vælger en topping. Når toppingen så er valgt, bliver kunden vist med en oversigt over sin cupcake, og kan herfra vælge at lægge kagen i indkøbskurven.

3. Validering af brugerrolle

Når en bruger logger ind, tjekkes der om den givede bruger har rollen “customer” eller “admin”. Er brugeren en admin, bliver brugeren videresendt til en dedikeret admin homepage, hvor man kan se alle brugerne i systemet samt deres ordre (og fjerne en given ordre). Er brugeren i stedet en customer, bliver brugeren videresendt til en dedikeret bruger homepage, hvor man kan vælge sin kage, lægge den i indkøbskurven, se sine personlige oplysninger samt konto, købe sin cupcake og også se alle sine egen ordre.

4. Brug af UUID

Hver gang der bliver oprettet et User-objekt, bliver det tildelt en tilfældig UUID variabel, som kunne se sådan her ud: 437927da-fa6b-46dc-9595-0cc1393a09b5. Det samme gælder sker hver gang der bliver oprettet et Order-objekt eller et Cupcake-objekt. De får også tildelt et UUID, som bliver smidt ind i databasen, så man nemt kan identificere det gældende objekt.

Status på implementationen

User story	Status
1	Tjek. Man kan lave sin egen cupcake, bestille og se status på ordren.
2	Tjek. Man opretter sig som det første og kan se alle sine ordre.
3	Tjek. Kunden kan endda selv indbetale penge til sin konto. Dog er denne feature ikke færdig, så lige nu kan kunden sætte papirspenge ind uden brug af kreditkort. Dette skulle meget gerne virke på et

	tidspunkt i fremtiden.
4	Tjek. Man kan se mange informationer i indkøbskurven.
5	Tjek. Som kunde kan man se sin mail inde på sin kontoside.
6	Tjek. Administratoren kan se alle kunder og deres respektive ordre.
7	Tjek. Administratoren kan se alle kunder i systemet under fanen “Kunder”.
8	Fifty-fifty. Jeg har valgt at gemme Cupcake-objekterne i en ArrayList i stedet for et Map, så man kan slette alle tingene i indkøbskurven, men ikke hvert enkelt. Så skulle man have valgt et Map i stedet.
9	Tjek. Man kan fjerne en kundes ordre.

Proces

Til at starte med smed jeg databasen i luften, lavede et schema og begyndte at implementere de forskellige features. Jeg valgte ikke at lave én user story ad gangen, men at opstille nogle features ud fra de user stories, der var givet. Jeg fandt dog hurtigt ud af, at de tabeller, jeg havde oprettet i mit schema skulle revideres et par gange igennem processen, da jeg senere tildelte brugeren en balance samt tildelte hver cupcake et id. Det blev i sidste ende en noget agil udviklingsproces, hvor jeg måske havde ønsket mig en lidt mere slavisk gennemgang. Jeg startede f.eks. med at give hver kundes ordrelinier i indkøbskurven et ID og smække hver en cupcake ind som en ny ordre. Det virkede skide godt, men jeg fandt senere ud af, at det selvfølgelig gav bedre mening at have én ordre for hver ordre, og hvor hver ordre havde x antal cupcakes tildelt.

Jeg arbejdede både på en bærbar og en stationær, så havde jeg gjort mere ud af Git, var overgangen mellem de to computere blevet meget bedre. Jeg burde også have givet hver feature en seperat branch og merget dem sammen igen for at skabe et bedre overblik over udviklingsprocessen.

I bund og grund var det en meget lærerig opgave. Meget på programmeringsfronten men endnu mere på procesfronten. Jeg er blevet overbevist om, hvorfor det er vigtigt

at gøre sit forarbejde og dele en udviklingsproces op i forskellige stadier/sprints. Det skaber et meget bedre overblik over processen.