CupCake Report

18.December 2018

Mohammad Hariri, Patrick Juul Diekmann, Andreas Heick, Rasmus Jarnborg Friis

CPHBUSINESS LYNGBY

Andreas = cph-ah384@cphbusiness.dk, Github = grem848

Mohammad = cph-moh682@cphbusiness.dk, Github = moh682

Patrick = cph-pd66@cphbusiness.dk, Github = Dieky

Rasmus = Cph-rf43@cphbusiness.dk, Github = Hoppedyr

Indledning

Cupcake er et produkt i form af en hjemmeside hvor en person skal kunne oprette en bruger, bestille cupcakes og se sin shopping historik. Rapporten beskriver forhold i databasen samt indeholder navigation og sekvens diagrammer. Der vil være et afsnit med fejl og mangler som er vigtigt at læse (status på implementation).

Baggrund

Virksomheden ANDY'S CUPCAKES er en ny virksomhed som sælger cupcakes på business-to-consumer markedet. Virksomheden er en produktionsvirksomhed da de selv laver deres cupcakes, og en detailvirksomhed som sælger Cupcakes direkte til forbrugerne.

ANDY´S CUPCAKES afsætning har været faldende siden begyndelsen, eftersom deres leveringsmetoder har anskaffet dem en del utilfredse kunder fordi der har været problemer med at deres leverandør simpelthen har taget en bid af folks cupcakes.

Derfor har virksomheden investeret i et helt nyt produktionsanlæg som kan lave cupcakes på under 1 min. Så derfor har de valgt at rebrande sig selv, så i fremtiden skal kunderne selv bestille cupcakes på deres hjemmeside og afhente dem i nærmeste butikker.

ANDY'S CUPCAKES krav til deres ny hjemmeside er følgende

1. kunderne har hver en konto hos butikken, og ordrer kan kun placeres, hvis kontoen har nok penge til at dække prisen.
2. Betaling håndteres af et andet system, og fra nu af skal indbetalinger håndteres manuelt i databasen, men udbetalinger sker, når cupcakes bestilles.
3. For at kunne betale med deres konto skal kunderne bruge et brugernavn og et kodeord til at logge ind, før de bestiller.
4. Cupcakes har en bund og en topping, som kan kombineres på mange måder, men en Cupcake skal altid have både bund og topping

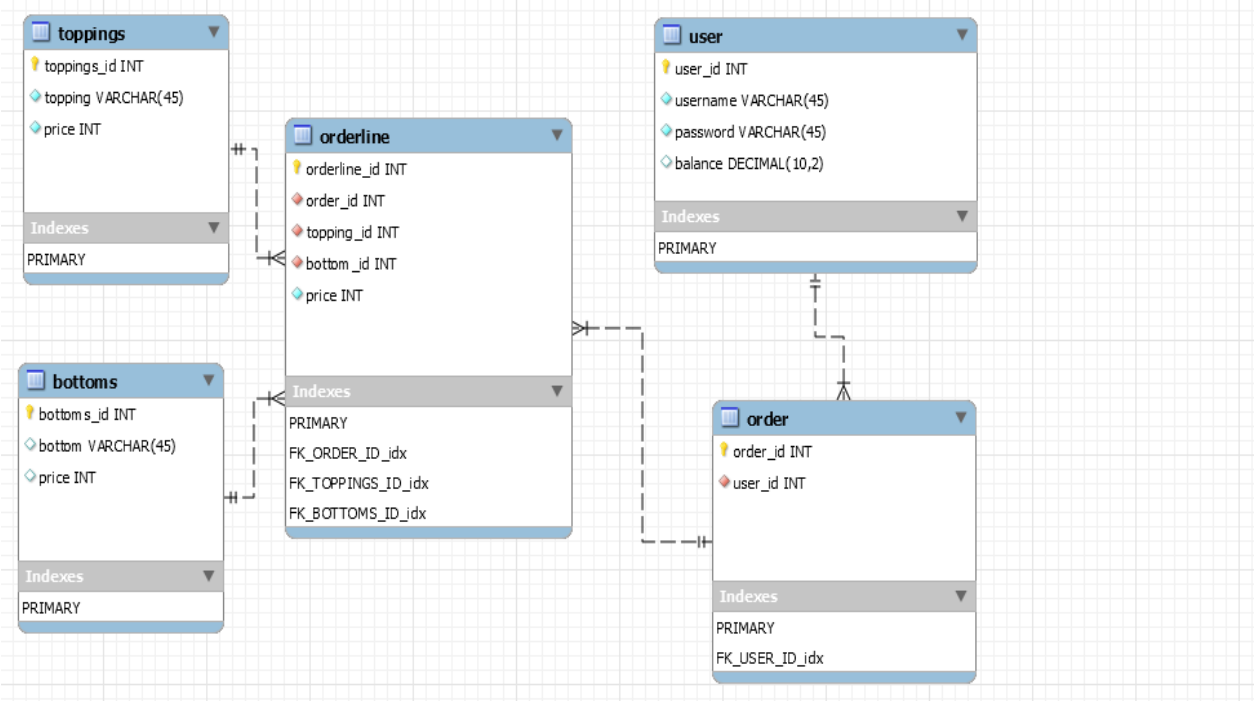
Teknologivalg

Disse teknologiske værktøjer er benyttet i programmet:

1. Netbeans IDE 8.2
2. MySQL Workbench 6.3 CE
3. Tomcat 8
4. Digital Ocean, Ubuntu 16.04.4 x64, 1gb Memory, 1 vCPU, 25gb SSD DISK, 1TB TRANSFER.
5. https://www.planttext.com/ (til at lave diagrammerne)

ER diagram

https://gyazo.com/09085c2f82984f6c114aa51fbf6aa3e4 ER diagram screenshot

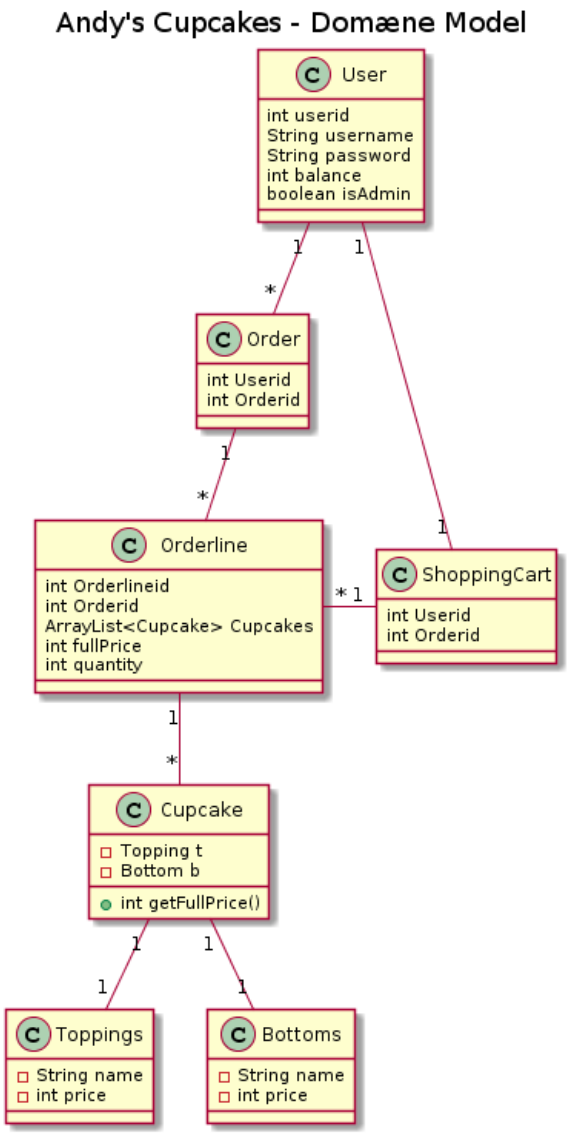


Til Orderline tabellen har vi foreign keys til toppings_id, bottoms_id og order_id. Da vi vil sikre os at en orderline bliver oprettet med den rigtige data. Alle foreign keys er forbundet til primary key i deres respektive tabel.

Order tabellen har foreign key til user som også er forbundet med foreign key til primary key. På den måde kan vi nemt hive data ud fra brugerens ID og se deres shopping historik.

De constraints der er sat op med foreign keys sørger for databasens integritet.

Domæne Model



Domæne model over Andy's Cupcakes, som viser en model over planen af hvordan systemet burde indrettes.

En User laver en Order, som laver en Orderline med produkterne (Cupcakes) i en liste.

Orderid, Userid og Orderlineid, bruges til at finde frem til en kunde, så de ikke forveksles med andre.

Hvis en ny Cupcake tilføjes lægges den i listen, og quantity på den bestemte Cupcake sættes til 1, tilføjes flere Cupcakes sættes quantity derefter, så brugeren kan se antal Cupcakes.

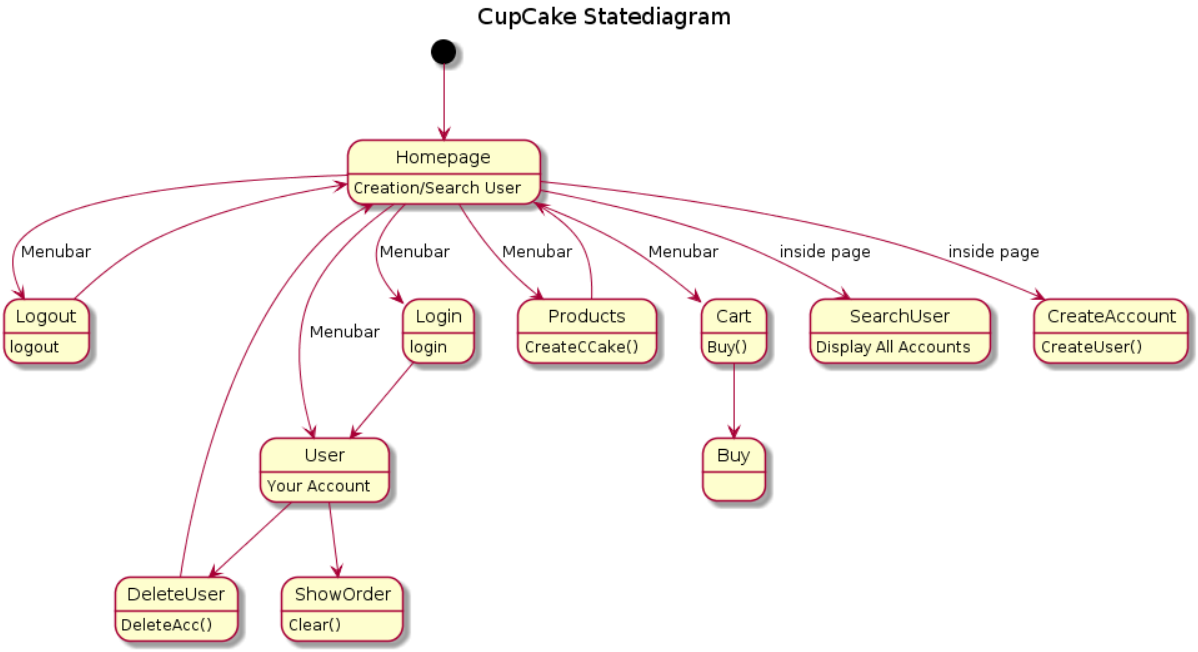
Med fullPrice ses den fulde pris af Userens Order. I ShoppingCart kan User se sine Orderlines, disse kan senere også lægges i en OrderHistory klasse, som dog ikke er med i denne model.

Navigationsdiagram

Det som brugeren oplever, er en række websider, hvor man kan indtaste oplysninger og navigere videre til andre sider. I større systemer kan det være svært at bevare overblikket over hvilke sider der er, og hvordan man kommer fra den ene til den anden. Navigationsdiagrammet er beregnet på at vise dette på en mere overskuelig måde. Som led i beskrivelsen af navigationsdiagrammet skal følgende med:

- Oversigtsdiagrammet. Hvis det bliver for stort må man dele det op. Men det er vigtigt at der er et overordnet diagram.
- Hvis man har benyttet sig af en "fælles navigations bar" i toppen af alle sider, skal man forklare det.
- Hvis nogle sider kun kan nås af nogle brugere (dem der har konto, dem der er logget ind, dem der arbejder i butikken,…), så skal det fremgå.
- Navne på jsp sider skal fremgå, og hvilke servlets der bringer en fra den ene side til den næste.

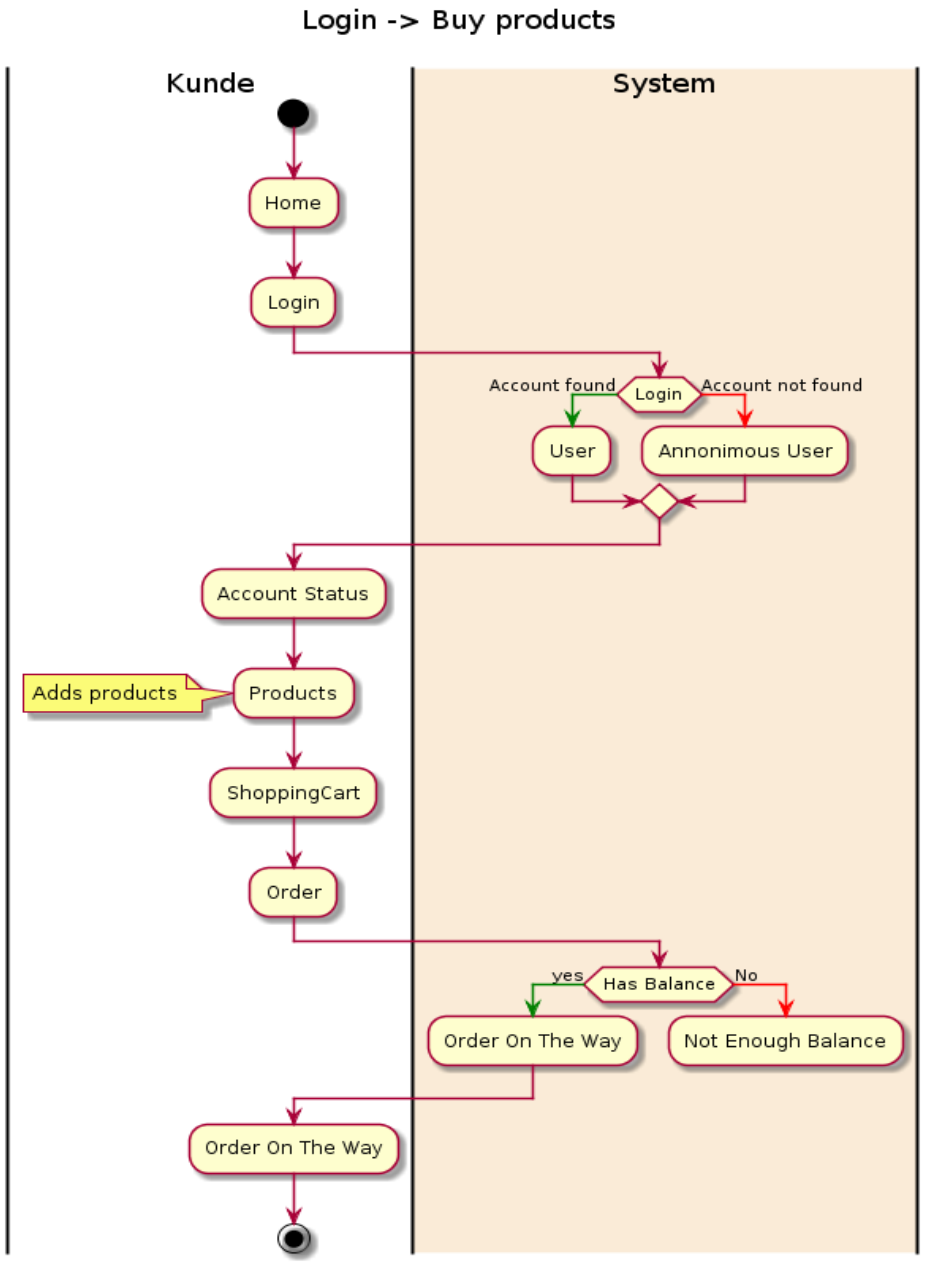
Navigationsdiagrammer laves som UML Tilstandsmaskiner state charts. Bemærk specielt at det, der hedder "composite state", er meget velegnet til at indkapsle at man er logget ind.



kundens første page er Homepage. Fra start har man en menubar, som man kan dirigere sin retning imod. Diagrammet fremviser det som ligger på menubaren. betydningen af inside page skal fortolkes således, at SearchUser og CreateAccount bliver videreført fra knapper i homepage, de ligger ikke på menubaren.

User kræver ikke at man er logget ind, for at bliver ført til siden. Login muligheden på hjemmeside er optionel, da vi giver kunden lov til at navigere og se produkter inden behøvet for at registrere sig på siden. dog kræver Shoppingliste (Cart), at man er registreret på siden, for både at se sin ordre og gennemføre købet.

Aktivitetsdiagrammer



Kunden kan have flere formål når han eller hun klikker sig ind på hjemmeside, blandt de formål kan forhåbentlig være at købe produkter. Kunden ankommer på siden og logger in. Afhængig om Kunden findes i databasen eller ej, vil følgende ske. Hvis kunden tilfældigvis ikke har en account, vil han være logget ind som en anonymous user. En anonymous user vil ikke have noget balance. Login vil medføre til Account Status hvor man vil have mulighed for at adde credit og forhøje sin balance. Kunden ankommer til Products, hvor han bygger sine cupcakes.

Efter det vil kunden ankomme til Shoppingcart og bestille sine ordre, hvis kunden har nok credit eller balance, så vil bestillingen blive gennemført med det samme. Hvis kunden ikke har nok balance vil den ikke tage imod købet og stå af.

Sekvensdiagrammer

Opgaven til onsdag

Et sekvensdiagram der starter med at en bruger har valgt en cup-cake, og nu vil købe den. [https://datsoftlyngby.github.io/dat2sem2018Spring/Modul2/Week4-Report/]

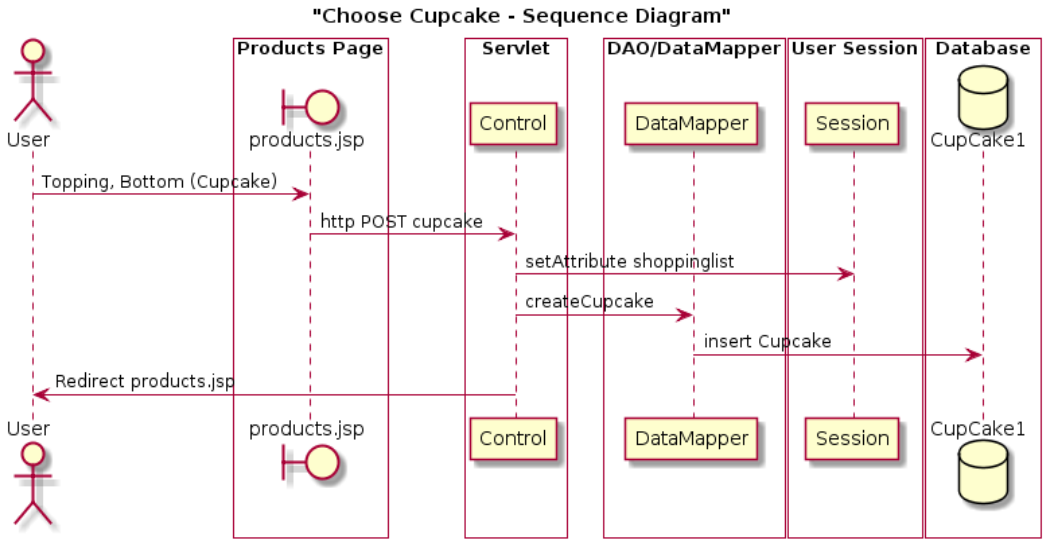
Begrundelse for 3 diagrammer:

For at simplificere og give overblik over den stillede opgave med sekvensdiagram, har vi valgt at dele den op i tre diagrammer.

1. Det første diagram er “Vælg en cupcake” diagrammet, som viser hvad der sker, når man vælger en cupcake.
2. Det andet diagram er “Vis shopping cart” diagrammet, som viser hvad der sker, når man vil se sin shopping cart, så man kan købe sine cupcakes.
3. Og det sidste diagram er “Køb cupcake” diagrammet, som viser hvad der sker, når man vil købe en cupcake.

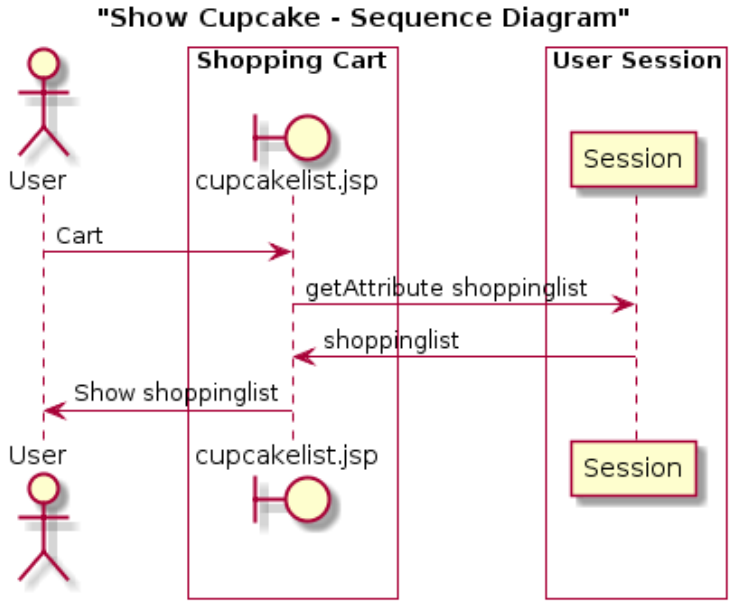
Hvis de tre diagrammer skulle laves som et, ville det blive meget stort og uoverskueligt vil vi mene, derfor har vi valgt at korte det ned i tre dele. Dette gør det også nemmere hvis man vil se på de individuelle dele af siden, og hvad de gør, uden at man skal sidde og finde det hele i et stort sekvensdiagram.

Vælg en cupcake - Sekvensdiagram



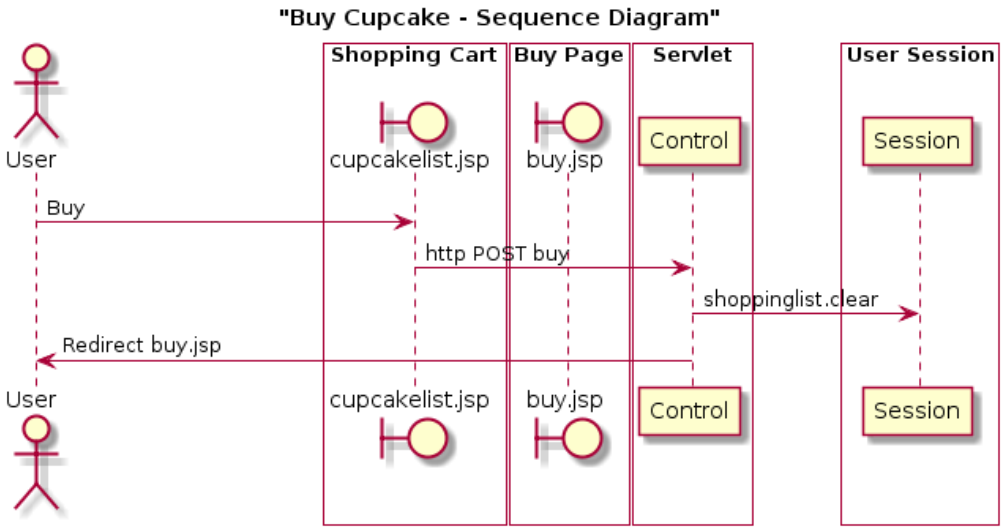
Til venstre har vi en bruger som på jsp siden products (produktsiden), i to bokse vælger henholdsvis en top og en bund til en cupcake. Derefter trykker brugeren på knappen “Create your cupcake”.
“Create your cupcake” sender et http POST “cupcake” kald til Control (Servlet) der bruger metoden setAttribute, til at smide cupcaken på brugerens session i arraylisten shoppinglist. Derefter kalder Control på createCupcake metoden i DataMapper, som laver en top og en bund, til et Cupcake objekt.
Cupcake objektet bliver så lagt ind i CupCake1 MySQL databasen, med en “insert into cupcakes” kommando.
Derefter returnerer Control et redirect til jsp siden products, der viser brugeren siden med produkter igen.

Vis shopping cart - Sekvensdiagram



Til venstre har vi en bruger som trykker på knappen “Cart” på navigationsmenuen i toppen af alle sider. “Cart” sender et redirect til jsp siden cupcakelist (shopping cart), som bruger getAttribute metoden til at returnere arraylisten shoppinglist, der indeholder de cupcakes brugeren har sammensat på jsp siden products (produktsiden).
Brugeren ser derefter sin shopping cart på cupcakelist jsp siden.

Køb cupcake - Sekvensdiagram



Til venstre har vi en bruger som trykker på knappen “Buy” på jsp siden cupcakelist (shopping cart). “Buy” sender et http POST “buy” kald til Control (Servlet) der tømmer arraylisten shoppinglist, som indeholder alle de cupcakes man har købt på brugerens session.
Derefter returnerer Control et redirect til jsp siden buy, der viser brugeren at købet er gennemført og ordren er klar til afhentning.

Javadoc

Der er lavet javadoc for sekvensdiagrammet Vælg en Cupcake (øverste).
Vi har derfor lavet Javadoc dokumentation for klasserne Control, DataMapper og Cupcake, som beskriver hvad de forskellige klassers metoder gør og skal.

Link til github Javadocs:
https://grem848.github.io/CupCakeShop/

Særlige forhold

Informationer gemt i Session

User; dette er den information som den loggede ind user har.
ToppingsList; dette er en ArrayList af alle de mulige toppings.
BottomsList; dette er en ArrayList af alle de mulige bottoms.
OrderList; dette er en kopi af shopping list, som bliver brugt til ordrehistorik.
users; dette er en liste af alle de users som er blevet lavet.
ShoppingList dette er en arraylist, som gemmer alle ens cupcakes man har lavet, og vil købe. Den bliver cleared hver gang man klikker buy.

Sikkerhed, validering & brugertyper

Sikkerhed

Vi sikrer vores brugere ved at bruge validering, af det brugeren prøver at logge ind med. Den måde vi valider brugers login er ved hjælp af en metode som vi har kaldt validateUser. Denne metoder tjekker på om det brugerne har skrevet ligger i vores database, fordi vi i vores program, når en User bliver lavet, gemmer denne User i databasen. Det vil sige at hvis vores validataUser ikke kan finde den bestemte bruger, så er han slet ikke blevet lavet. Hvis en User ikke bliver fundet i vores database bliver brugeren i stedet logget ind som anonymous user, med en fejlbesked.

Validering

Meget af vores validering sker i MySQL, da databasen er sat op til at håndtere null. I databasen er det sat op sådan, at kun balance og admin er sat til at acceptere null, ved at de sættes til 0 som standard i tilfælde af null. Ellers er programmet bygget sådan at man umiddelbart kan lave det input man vil.

Brugertyper

Vi har i vores program kun to brugertyper, admin og bruger (ikke admin), men begge er kun håndteret ved en boolean, da vi ikke har fokuseret på at lave administrator redskaber, på vores hjemmeside.

Status på implementation

En af det store ting som vi ikke har fået fixed, er at når vi henter ting fra vores database benytter vi kun PreparedStatement på vores validateUser. Hvilket betyder at alle steder hvor vi har user input kan man lave sql injection.

En anden ting som vi mangler at fixe er at vi ligger en færdig CupCake i databasen, men vi bruger den ikke til noget endnu. Så den måde vi laver en ordrehistorik er ved hjælp af en ArrayList i sessionen, hvilket betyder at brugerne ikke har deres egne personlige historie, men kan se alle brugernes historik.

Dette skyldes at vores nuværende database ikke er oprettet efter ER diagrammet. Integriteten i databasen er derfor ikke på plads. Der skal derfor laves/redigeres CRUD operationer når den nye database bliver oprettet og overtager.

JSP implementation

Umiddelbart har vi alle de jsp sider vi skal bruge, de er bare ikke fuldt implementeret eller lavet til minimum. En side vi mangler er dog en side, og en håndtering af fejlfuld login, som lige nu bare logger en ind som en anonymous user, som virker dårligt, hvis overhovedet, på alle sider.

Side styling

Vi har nået at style alle sider, men der er altid plads til forbedring. En bedre pladsudnyttelse er også noget vi kunne bruge, da alt sker i et slags "tårn", med bokse i midten. En bedre måde ville være at have menuer, eller knapper og information i højre og venstre side, så man udnytter pladsen bedre.

Generelle fejl og mangler

- Logout knappen virker ikke, den sender bare en til forsiden, uden at slette brugeren i sessionen.
- En planlagt opdatering knap på user.jsp (brugersiden), som skulle tillade ændring af username, password, balance (administrator) og admin (administrator). De to sidste kun af administratorer.
- Generelt mere på siderne, mange jsp sider som usercreated, user deleted, userupdated (ikke brugt) og buy, kunne optimeres og have mere information og interaktion.