OLSKER CUPCAKES

Marts 2020

Datamatiker 2. Semester, Hold E

Alexander Pihl
Email: cph-as509@cphbusiness.dk
Github: https://github.com/AlexanderPihl

Morten Rahbek
Email: *mail@hej.dk*
Github: *mail@hej.dk*

Mick Larsen
Email: *cph-ml616@cphbusiness.dk*
Github: *https://github.com/MivleDK*

Per Kringelbach
Email: cph-pk171@cphbusiness.dk
Github: https://github.com/cph-pk

Indhold

1. Indledning

Olsker Cupcakes er en pengeløs webshop lavet primært i Java, MySQL og JSP med en *servlet.*
Siden er deployed på en *Tomcat webserver* hostet på en droplet hos Digital Ocean.
Til programmet hører en database udarbejdet i MySQL med *Workbench* samt nærværende dokumentation.

Projektet tager udgangspunkt i undervisning modtaget på datamatikerstudiet, andet semester i marts 2020. Projektet afspejler ca. 3 ugers arbejde af en gruppe på fire og er resultatet af undervisningen i backend, frontend og webstack.

2. Baggrund

Olsker Cupcakes (Herfra benævnt "Virksomheden") er en lille, bornholmsk forretning som har startet et økologisk iværksættereventyr. Virksomheden sælger cupcakes efter kundespecifikation – dvs. kunden "Bygger" selv sin cupcake bestående af en bund og top ("Topping").

Virksomheden har stillet krav om en lille webshop (Uden implementering af et betalingssystem) hvor man skal kunne registrere sig og herefter logge ind som kunde.
Som kunde skal man kunne "Bygge" sin egen cupcake ved at vælge en bund og top samt antal som lægges i en indkøbskurv. Kunden skal herfra kunne sende sin samlede bestilling, akkurat som i en webshop.

Da der ikke implementeres et betalingssystem skal virksomheden i stedet kunne administrere kredit og kunder igennem en administrationsside. Desuden skal virksomheden have mulighed for, at administrere og holde styr på de enkelte ordrer.

Umiddelbart kan man betragte projektet som en lille, men skalerbar, pengeløs webshop.

3. Teknologivalg

Projektet er bygget i Java, på Windowsmaskiner i en Maven projektstruktur.
Programmet er designet efter en "*Command pattern"* designskabelon med brug af *servlets*, en *frontcontroller* og en *command* class for at gennemføre command pattern løsningen. Hertil er der lavet klasser der understøtter de respektive funktioner på .jsp-siderne.
Hele projektet er versionsstyret med git på Github. "Milepæle" er merget til en master-branch imens løbende udvikling er styret på en Production-branch.

Der er desuden udarbejdet en omfattende dokumentation i javadocs som kan ses på projektets github side.

Følgende software samt teknologier er anvendt:

Programmering og syntaks

- V. 1.8 Java
- V. 8.0.18 MySQL
- Javascript (ECMAScript 2015)
- HTML og HTML5
- JSP
- CSS3
- Syntaks: PlantUML

Frameworks og teknologier

- V. 4.12 JUnit (Ikke implementeret)
- V. 8.0.19 MySQL Connector Java
- V. 4.3.1 Bootstrap
- V. 7.0 JavaE Web API
- V. 1.2 JSTL

Software

- V. 2019.3.4 IntelliJ IDEA (IDE)
- V. 8.0.18 MySQL Workbench
- Hhv. Google Chrome (v. 80) samt Firefox (v. 74)

4. Krav

Følgende User Stories er blevet opstillet i samarbejde med kunden:

User story 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.

User story 2
Som kunde kan jeg oprette en konto/profil for at kunne betale og gemme en en ordre.

User story 3
Som administrator kan jeg indsætte beløb på en kundes konto direkte i MySql, så en kunde kan betale for sine ordrer.

User story 4
Som kunde kan jeg se mine valgte ordrelinier i en indkøbskurv, så jeg kan se den samlede pris.

User story 5
Som kunde eller administrator kan jeg logge på systemet med email og kodeord. Når jeg er logget på, skal jeg kunne min email på hver side (evt. i topmenuen, som vist på mockup'en).

User story 6
Som administrator kan jeg se alle ordrer i systemet, så jeg kan se hvad der er blevet bestilt.

User story 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.

User story 8
Som kunde kan jeg fjerne en ordre fra min indkøbskurv, så jeg kan justere min ordre.

User story 9
Som administrator kan jeg fjerne en ordre, så systemet ikke kommer til at indeholde udgyldige ordrer.
F.eks. hvis kunden aldrig har betalt.

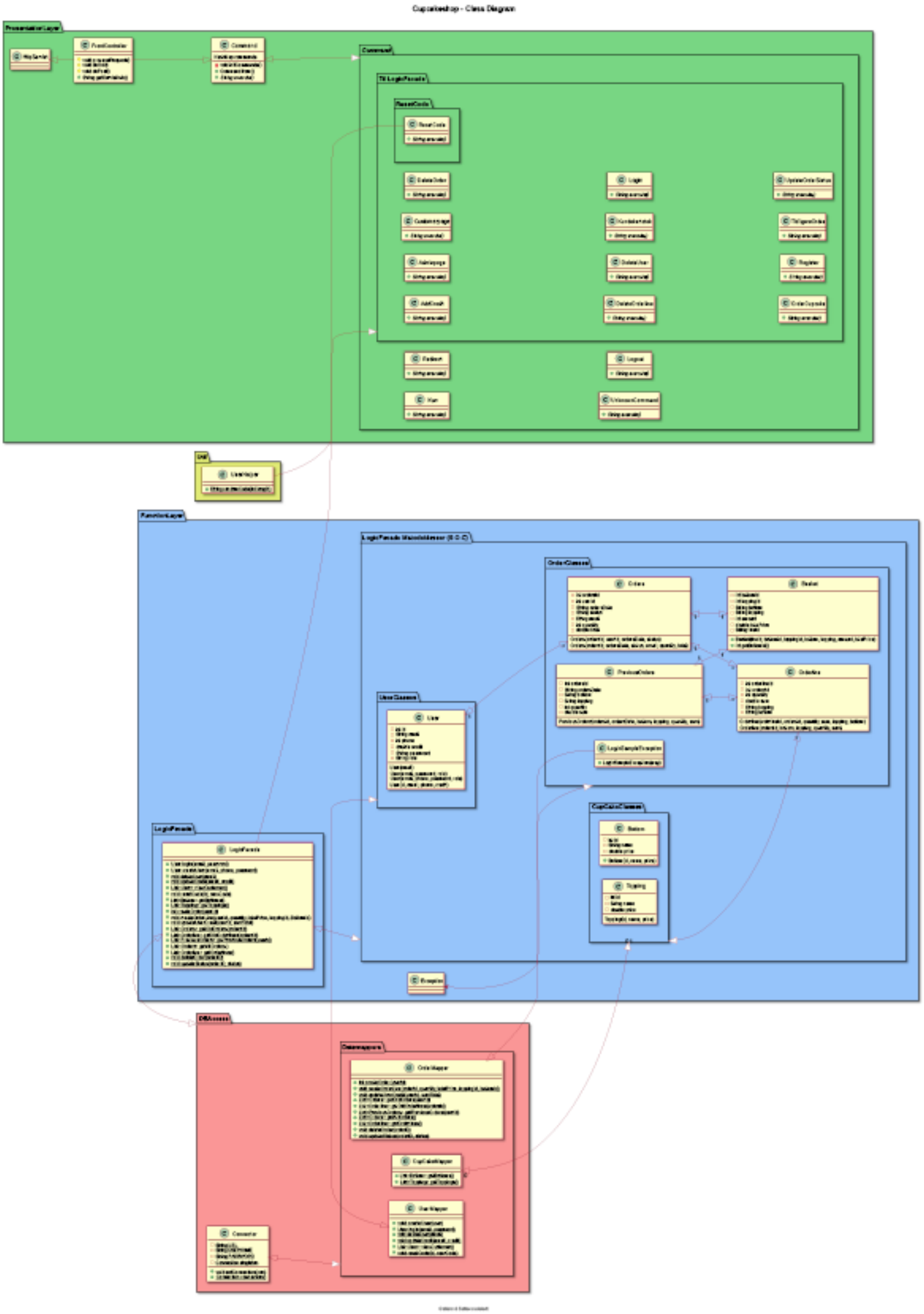
5. Diagrammer

5.1 Domænemodel

(Vedhæftes denne fil)

UML.pdf

Domænemodellen afspejler, at næsten alle operationer på siden, går igennem frontcontroller, command og til den relevante klasse.

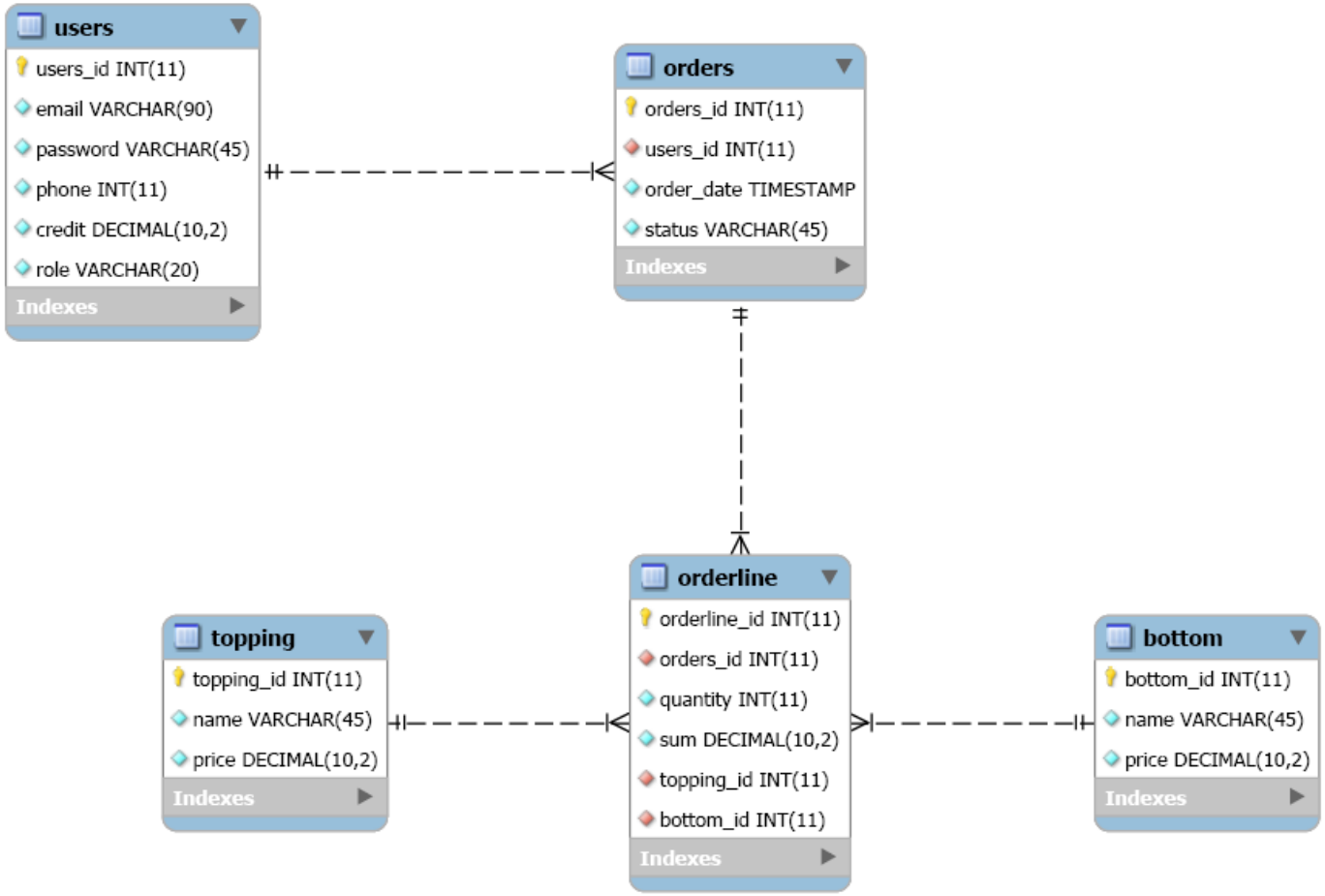


5.2 ER diagram

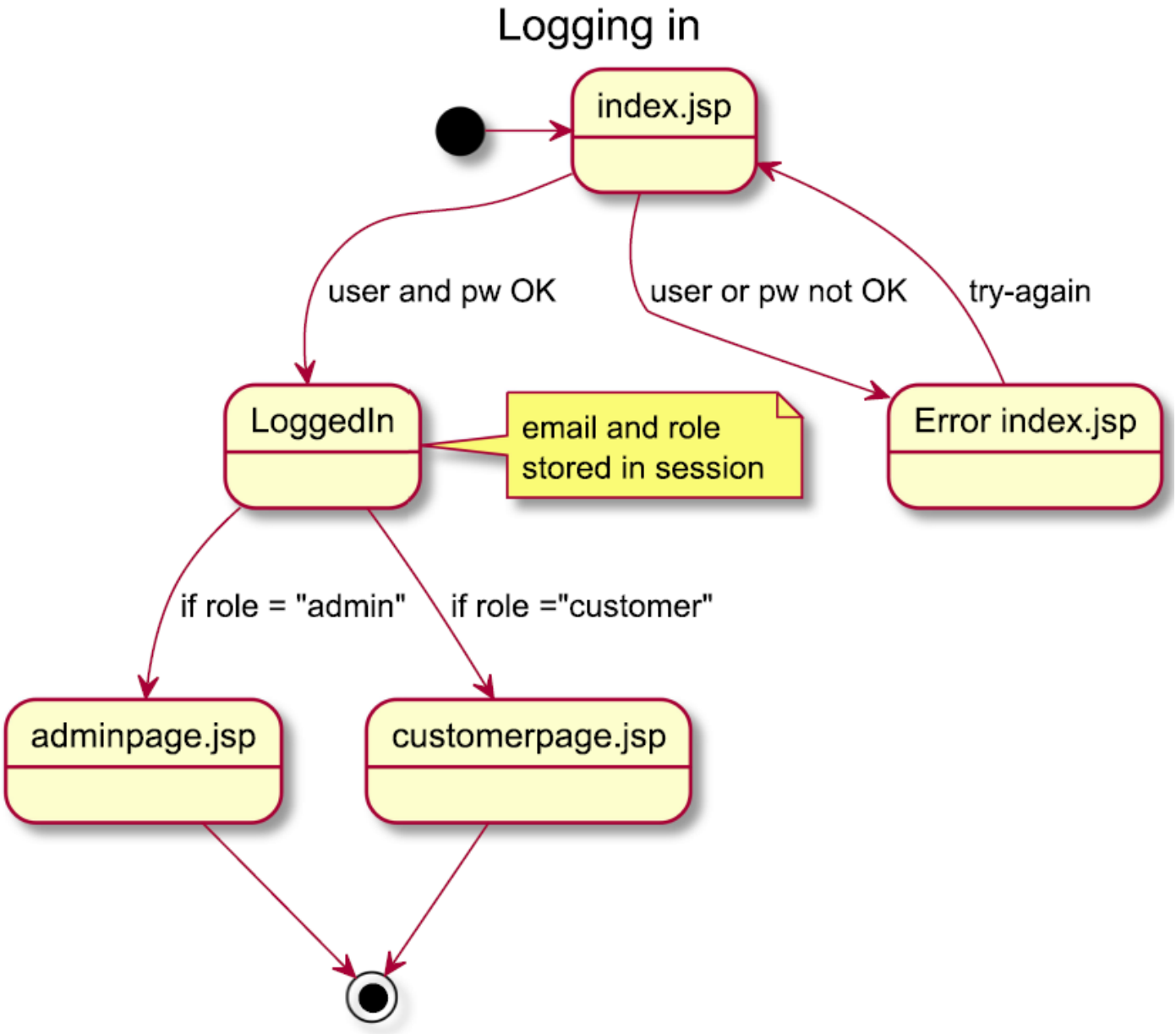
Nedestående diagram er en illustration af hvordan databasen er opbygget.

Databasen er normaliseret til 3. normalfor, dog med undtagelse af *topping* og *bottom* tabellerne. Disse kunne samles i en enkelt tabel således at, hhv. id'et var unikt for at undgå to tabeller med hhv. pris og navn i begge.

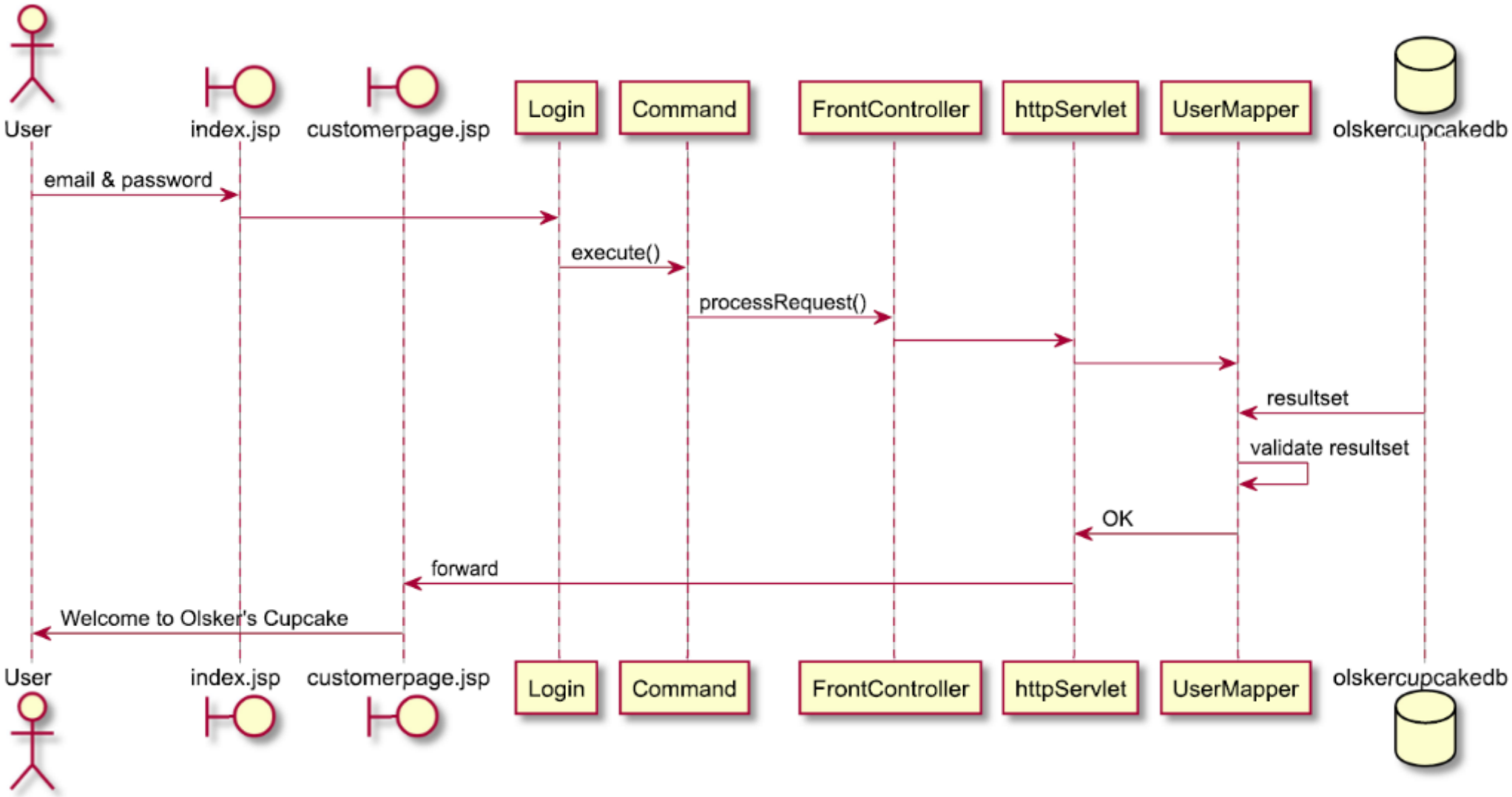
Det er dog besluttet at undlade denne normalisering her, da separationen af hhv. *toppings* og *bottoms* giver god mening ift. måden der interageres med disse i programmet.



5.3 Navigationsdiagram for login



5.4 Sekvensdiagram for Login



6. Særlige forhold

6.1 Sessionen

Programmet benytter ikke cookies til at gemme oplysninger om login, basket, tracking eller andet.
I stedet gemmes informationer i den enkelte information.

Når brugeren logger ind, initialiseres *Lists* (Som hentes fra databasen) allerede på loginsiden.
Initialiseringen sker i loginformen forsiden således at listerne er tilgængelige på den side brugeren sendes til. Sessionen gemmer derfor information om:

- List: (Cupcake) bottom
    - Cupcake bottom navne, priser og id
- List: (Cupcake) topping
    - Cupcake topping navne, priser og id
- Objekt: User
    - Hvem er logget ind med tilhørende information såsom role, kredit, email, mm.
- Objekt: Basket
    - Hvilke *orderLines* er lagt i kurven

6.2 Exception handling

Programmet anvender exceptions flere steder.
Foruden gængse exceptions såsom *MySQLException* er følgende to extensions implementeret:

- LoginSampleException
    - Anvendes ved *throws* i forbindelse med fejl på loginsiden. Dette gælder for både login og registrering.
- UnknownCommand håndterer eventuelle exceptions som opstår på alle andre sider end loginsiden.

6.3 Validering

Validering foretages primært på to forskellige måder.

Bootstrap har visse indbyggede valideringer i fx inputfelter defineret som ”email”. Disse forhindrer brugeren i at angive en, syntaksmæssigt, forkert email (Uden ”@”).

Programmet har desuden validering på registreringen af nye brugere. Det sikrer at alle informationer skal tastes før en bruger kan oprettes.

Der findes ingen øvre grænse for, hvor mange cupcakes brugeren kan bestille, men det er dog ikke muligt at bestille et negativt antal.

6.4 Sikkerhed og login

Det lokale miljø
Programmet er udviklet lokalt på localhost. Derfor har sikkerheden omkring databasen været lav. Superbrugeren til databasen er root, password har været forskellig afhængig af hvilken maskine programmet er afviklet på.

Deployment
For deployment på Digital Ocean bør sikkerheden omkring databasen højnes.
Dette kan bl.a. opnås ved at lave en ny databasebruger som kun har de relevante CRUD rettigheder og et stærkt password.

Generel sikkerhed
Der er ikke indkøbt domænenavn til siden, hvorfor den pt. ligger på en ip-adresse.
Desuden er hverken SSL/TSL-certifikater implementeret hvorfor HTTPS ikke kan anvendes.
Der stilles heller ikke krav om længde eller styrke af nye brugeres password hvilket selvfølgelig bør implementeres.

Metodekald
Programmet er designet efter en command pattern designskabelon. I dette henseende skal operationer fra frontcontroller klassen igennem command klassen hvorefter den efterspurgte funktionalitet kaldes i en relevant klasse.
Operationer som har kald i databasen går alle igennem klassen *LogicFacade* som håndterer al kommunikation med databaseklasserne. Dette er gjort for at opnå en god *separation of concerns*. Det er dog pt. muligt at lave kald direkte ned til databaseklasserne, men frarådes på det kraftigste for at overholde formål og struktur med programmets design.

6.5 Brugertyper i databasen
Som nævnt i afsnit seks er brugeren på et lokalt testmiljø op til den enkelte udvikler.
I en online, deployed version vil der være én databasebruger restricted til, udelukkende at have de absolut nødvendige rettigheder ifbm. CRUD operationer.

6. 7 Andre forhold

Websitet er designet med Bootstrap v. 4.3.1.
Bootstrap er ikke hentet lokalt på serveren, men hentes via CDN.

Desuden har få ændringer af layoutet været nødvendigt, hvorfor der er oprettet en *style.css* i *includes*-mappen, i projektstrukturen.

7. Status på implementation
Webshoppen er 100% driftsklar og implementeret på en droplet, hostet hos DigitalOcean.
Alle user stories er implementeret iht. kundens krav.

7.1 Oprydning i CSS

Få elementer er stylet direkte på jsp-siden og bør styles på enten ID eller klasseniveau i den centrale *style.css*.

7.2 Flere, beskrivende exceptions

Flere exceptions kunne med fordel implementeres, eksempelvis hvis brugeren forsøger at bestille et negativt antal cupcakes.

7.3 Generering af passwords

Foruden kundens krav er en nulstilling af brugerpasswords implementeret.
Denne nulstilling genererer et tilfældigt password til brugeren, i tilfælde af, at de har glemt deres eksisterende. Implementeringen af lavet i en package ved navn ”util” i klassen *UserHelper*.

Unit tests

Unittests med jUnit er ikke implementeret, men dog importeret i maven projektstrukturen.

8 Konklusion

Hjemmesiden til projektet Olsker Cupcakes er 100% implementeret efter kundens krav og i øvrigt gode retningslinjer for programmering, hvad anger design patterns, semantik og googles styleguide for Java så vidt muligt.