CUPCAKE-RAPPORT

PRØVEEKSAMEN MARTS 2019 – B KLASSEN

Gruppe:
Ahmed Othman, cph-ao141@cphbusiness.dk, bananahowl
Frederik Løwe Lassen, cph-fl91@cphbusiness.dk, frederikll
Emil Skovbo, cph-es149@cphbusiness.dk, emil-skovbo
Frederikke Nilsson, cph-fn72@cphbusiness.dk, fred8728

Projekt udarbejdet i perioden: 25/02-19 – 10/03-19
Rapport udarbejdet i perioden: 11/03-19-17/03-19

Indhold

Indledning

Denne rapport omhandler udviklingen af en webshop til et cupcake-bageri. Først og fremmest for, at give et bedre overblik er der blevet udarbejdet diverse diagrammer, heraf ER-diagram over vores database, navigationsdiagram over webshoppen, samt to sekvensdiagrammer, hvoraf det ene beskriver oprettelsen af en bruger og den anden beskriver bestilling af cupcakes. Til sidst i rapporten har vi i afsnittet 'status på implementation', samlet fejl i projektet, samt mangler ift implementation. Derudover er det muligt gennem tests afsnittet, at se hvor stor en dækningsgrad testene har på programmet.

Baggrund

Vores kunde er et cupcake-bageri, som udelukkende er specialiseret inden for cupcakes. Bageriet adskiller sig fra andre bagerier, da de er indehaver af en cupcake-maskine, som kan bage en cupcake på et sekund, hvilket giver bageriet mulighed for at tilfredsstille deres kunder sekundet efter de har bestilt. Det er på baggrund af dette, at kunden ønsker en webshop, som giver deres kunder mulighed for, at bestille online og hente deres bestilling direkte i deres butik sekundet efter.

Kundens overordnede krav til systemet er:

- At kunden skal kunne oprette en bruger, samt logge ind.
- At kunden skal kunne overføre penge til sin konto for køb af cupcakes.
- At der skal kunne oprettes en ordre via. en bruger.
- At kunden skal vælge en bund og en topping, samt antal ved en bestilling.
- At kunden skal kunne modtage en faktura via email.

Teknologi valg

For, at tilgå projektet skal følgende software først og fremmest være installeret:

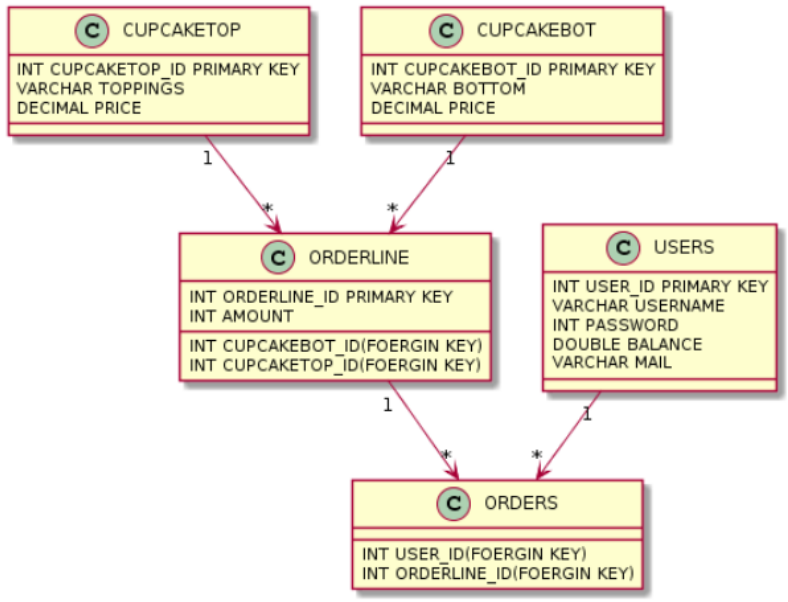
- Netbeans IDE 8.2
- MYSQL workbench 8.0 CE
- Tomcat apache - 8.5.38

Backend-delen er udarbejdet via mysql, en servlet, samt diverse java-classes/jsp-sider inde i Netbeans IDE 8.2. Dernæst har vi udarbejdet frontend-delen på den enkelte jsp-side ved anvendelse af html og css. Den primære html/css-kode befinder sig i en separat css-fil kaldt "style.css".

Krav

Webshoppen vil skabe værdi for bageriet, da kunden har mulighed for at vælge lige præcis den kombination af cupcakes som kunden foretrækker, udover det vil webshoppen give kunden muligheden for at bestille døgnet rundt - dog kan bestillingen først hentes ved åbningstid, hvilket også bringer os hen på visionen i forhold webshoppen. Cupcake-bageriets vision er, at gøre det muligt for alle cupcake elskere, at sammensætte den helt perfekte cupcake i ro og mag.

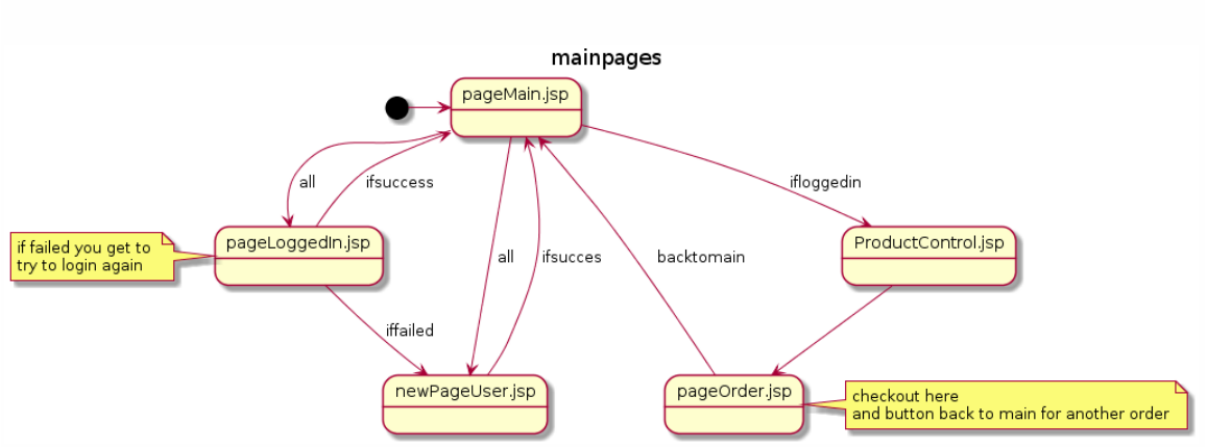
ER diagram



Ovenstående ses ER-diagrammet, som viser vores tabeller, samt relationer i databasen. Tabellerne er lavet, således at de er på 3 normalform. Vores tabeller har derfor hver især en primær nøgle, som identificerer en række i tabellen - et auto-genereret id, derudover har alle kolonner den samme datatype og ingen "multivalues". Tabellerne er ligeledes lavet, således at der ikke er nogen partielle afhængigheder, hvilket for eksempel ses i users og orders tabellerne. Disse to tabeller indeholder begge et user_id, dog er forskellen, at user_id i users er en primær nøgle og user_id i orders er en foreign key, hvilket medfører, at en ændring af users user_id vil ligeledes blive ændret i orders tabellen. Som det sidste for, at overholde 3. normalform er der blevet taget højde for, at der ikke er nogen transitive afhængigheder ved, at lave to tabeller i stedet for en. I dette tilfælde cupcakebot og cupcaketop, som vi har samlet i orderline og der vil derfor ikke opstå gentagelser i tabellen.

Overordnet ses fungerer vores tabeller, således at cupcakebot og cupcaketop, hver især indeholder et cupcakebot/top_id, smagsvarianter, samt en pris. Disse to tabeller bliver anvendt ved en ordrebestilling i orderline, hvor cupcakebot_id og cupcaketop_id, samt prisen bliver samlet - derudover er der ligeledes oprettet et orderline_id, således at alle ordre kan ses med et user_id og et orderline_id. User_id kommer fra users tabellen, hvor vi har en 1 til * relation.

Navigationsdiagram



Vi har en servlet(fremgår ikke i diagram) som kontrollerer hvilken JSP side man bliver sendt til. Heri har vi en switch(action) som styrer hvilke sider man kommer til.

Brugeren starter på vores pageMain hvor man har 3 muligheder: pageLoggedIn, ProductControl og newPageUser. newPageUser åbner siden hvor brugeren kan indtaste sine oplysninger og oprette et login til vores side. Dette skal i fremtiden være et must for brugeren, men lige nu kan brugeren godt købe vores vare uden at være logget ind. Hvis brugeren forsøger at oprette et login som allerede findes vil han få en fejl og blive bedt om at prøve igen. Når brugeren opretter et login som ikke er i brug vil det blive gemt i vores database, og brugeren vil blive sendt tilbage til pageMain, hvor brugeren kan vælge pageLoggedIn og indtaste sit nye login.

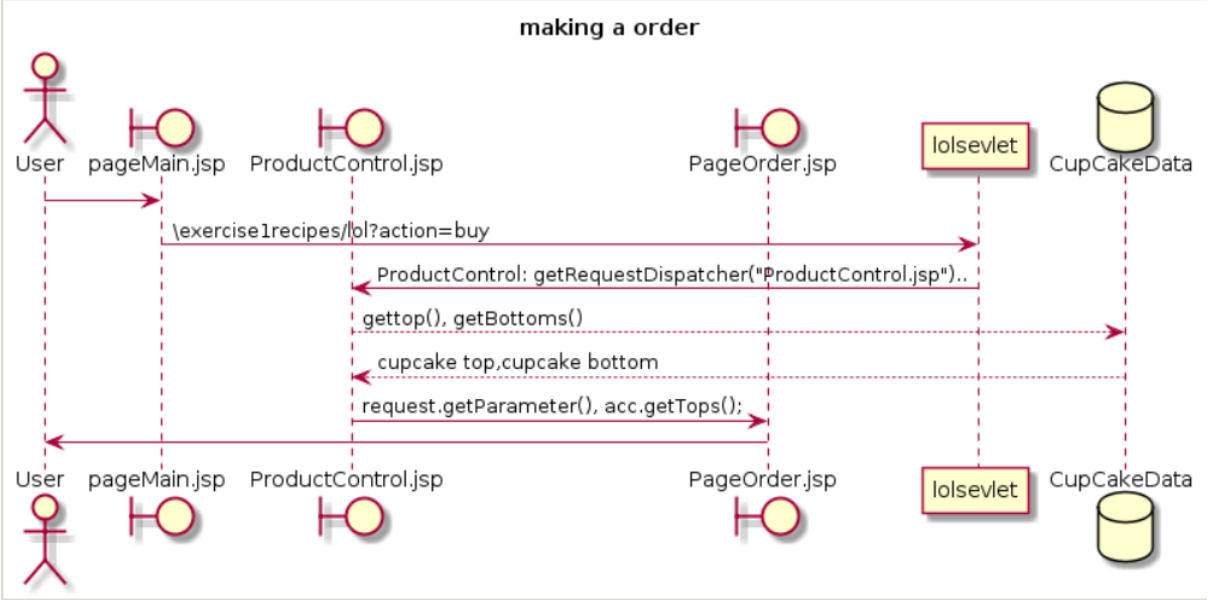
Hvis man taster forkert eller på anden måde indtaster et forkert login i pageLoggedIn vil brugeren få af vide at der var en fejl og man kan prøve igen, eller man kan vælge at blive sendt til newPageUser for at oprette en ny bruger.

Hvis man lykkedes med at logge ind med korrekte oplysninger bliver man sendt til pageMain(ProductControl session startes).

I vores ProductControl er der to dropdown lister, en med valg af tops og en med valg af bottoms, og herefter en quantity af hvor mange cupcakes man vil købe. Når man har valgt kan brugeren trykke 'add to cart' hvorefter orden bliver gemt i pageOrder. Fra pageOrder kan brugeren se sin ordre, pris, samt vælge at fjerne sin ordre og blive sendt tilbage til ProductControl. Udover det kan brugeren gå til checkout(bruger betaler og gemmes i database).

Sekvens diagrammer

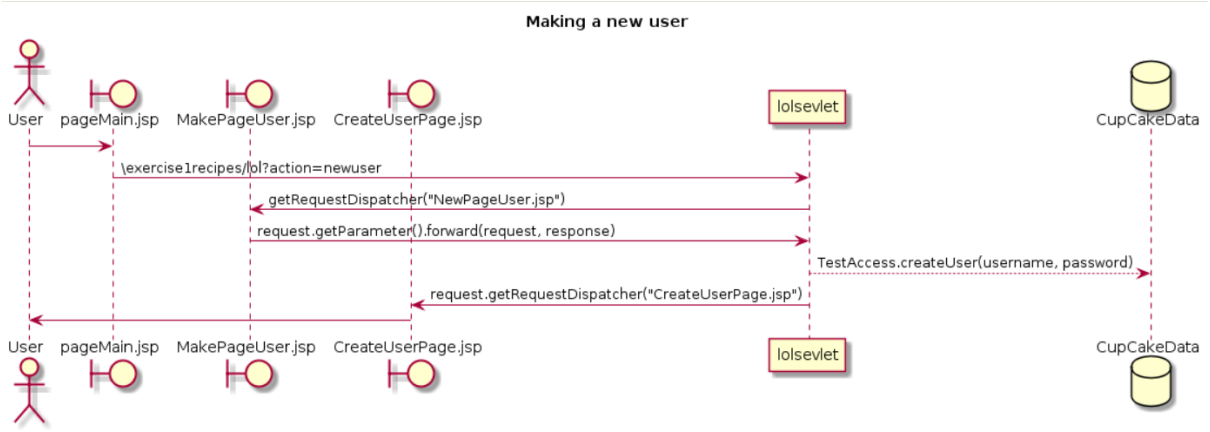
Making a order:



I sekvensdiagrammet for "making a order" ses processen for, hvordan en bruger opretter en ordre på siden. Brugeren starter inde på pageMain, hvoraf brugeren vælger "Buy something" knappen, som sender en reference til servletten med den givet action i referencen bliver metoden getRequestDispacher("ProductControl.jsp") kaldt i vores servlet ("lolservlet", frontcontroller). Herfra sender lolservletten brugeren til productControl-siden og her kræves nogle værdier for toppe- og bunde af cupcakes fra databasen til droptabellen. ProductControl.jsp kalder på metoderne getTop() og getBottoms(), som sender et sql statement til databasen, som så returnerer de ønskede værdier. Droptablene viser derefter de forskellige eksisterende smagsvarianter. Gennem ProductControl.jsp vælger brugeren dernæst sin ønskede cupcake kombination ved, at vælge en top og bund fra drop tabellerne, samt antallet af dem.

Når man er færdig og trykker på "bestil" knappen bliver det sendt videre til pageOrder.jsp. På denne side bliver der kaldt på flere request metoder, som tager værdierne for cupcakes toppe og bunde, samt deres priser. Derudover bliver der også brugt metoder, som gør at de modtagede værdier bliver sat ind, således at det bliver sat ind i ordren, samt en samlet udregnet pris for cupcakene - hvor at top og bund bliver lagt sammen og multipliceret med det valgte antal. Alt dette vises på PageOrder, således at brugeren kan se sin bestilling. Herfra er det muligt, at gå tilbage til pageMain med et link og ordren er nu lavet og klar til afhentning.

Making a new user:



Ovenstående ses sekvensdiagrammet for oprettelse af en ny bruger på webshoppen. Overordnet set kommer brugeren ind på pageMain.jsp og trykker på "create a new user", hvorefter man skifter fra pageMain til lolservletten. I den er der en switch case, hvoraf man ud fra en angivet action sendes til en bestemt jsp side. I dette tilfælde bliver actionen newUser brugt, som kalder metoden: getRequestDispatcher("NewPageUser.jsp") som sender brugeren til MakePageUser.jsp. På denne side kan brugeren indtaste sit ønskede brugernavn og password og når der trykkes på "lav bruger" knappen bliver man sendt tilbage til lolservletten. Lolservletten sender indtastede informationer fra .jsp siden til databasen som et SQL statement der indsætter det angivne username, og password i databasen og derefter gemmer brugeren i systemet. Dette er ikke synligt for brugeren, men bliver sendt til createUserpage.jsp. Derefter bliver brugeren bekræftet i, at det ønskede login og password nu er blevet instanseret og gemt i databasen.

Særlige forhold

Programmet er udviklet, således at vores cupcakes er dynamiske, i form af, at vi henter information fra vores database, som dermed viser de forskellige eksisterende smagsvarianter cupcake-bageriet har. Baggrunden for dette er, at man kun skal opdatere databasen og ikke ændre i koden hver gang der skal tilføjes en ny smagsvariant. Ligeledes bliver vores brugernavne, samt password gemt i databasen.

Exception:

I forhold til håndtering af exceptions har vi anvendt disse i form af en try-catch hver gang vi benytter os af Data_access klasses metoder, som alle er metoder der sender en query string til vores database. Dog havde vi tænkt os, at lave en pop-up besked til brugeren, som viser en fejlbesked/kode ud fra den enkelte fejl, hvilket for eksempel kunne være manglende balance på brugerens konto eller ugyldigt brugerinput.

Sikkerhed:
Sikkerhedsdelen er minimal på vores program, da en bruger kan lave det username og password de ønsker, således at det overholder 45 karaktere, hvilket kan være usikkert, da en bruger muligvis vil vælge et nemt password. Udover dette anvender vi hidden, således at vi gemmer parametrene fra URL'en.

Brugertyper:
I vores database har vi ingen brugertyper - kun brugere. I teorien ville vi gerne have haft to forskellige slags brugere, customers og admins. Admins skulle have adgang til JSP-sider, hvor de kunne redigere og styrer hjemmesiden, samt dens Customers. Customers ville være brugertypen til kunder og de skal kunne købe og bestille CupCakes.

Nedenstående ses jdbc driveren:

```
public DBConnector() throws Exception {
    Class.forName("com.mysql.cj.jdbc.Driver").newInstance();
    String url = "jdbc:mysql://" + IP + ":" + PORT + "/" + DATABASE;

    this.connection = (Connection) DriverManager.getConnection(url, USERNAME, PASSWORD);
}
```

Status på implementation

I dette afsnit ses hvilke fejl og mangler det vil være muligt, at finde i programmet.
Mangler:

- PageOrder.java mangler, at blive oversat til en jsp-side, samt at blive stylet.
- At oprette delete-funktion ift. CRUD-metoderne.
- Der har været flere metoder som ikke kunne blive implementeret.
- ER-diagrammets tabel user indeholder en kolonne med mail - denne er sat, som en default value, da vi ikke bruger den.
- Ingen validering af brugerinput.
- Vi mangler customer og admin page.
- Vi mangler at gemme invoices.
- Lige nu sender vi ordrer til databasen når kunden tilføjer til shoppingcart, ikke når de vælger 'checkout'.

Fejl i programmet:

- Vi får en error, når en bruger logger ind med brugernavn og kodeord, selvom brugernavn og kodeord er korrekt.
- PageOrder.java fungerer ikke, når man opstarter den separat. Dog fungerer det, når man tilgår den gennem webshoppen.

Andet:

- Grundet misforståelse af opgave udviklede vi webshoppen gennem java klasser og måtte derfor oversætte dem alle til jsp-sider, hvilket har været tidskrævende og skabt en del problemer - derfor indeholder programmet både jsp-sider, samt java klasser på det samme. Det har også gjort at der er blevet flere elementer som ikke er blevet implementeret pga. tidspresset.

Test

Der er udelukkende blevet lavet test på syntaksen i Data_access gennem vores main-class inden implementation af dem i servlet/java/jsp-sider. Vi vil derfor mene, at dækningsgraden af test på vores program er minimal.

Klasse:	Data_access
Metoder:	public ArrayList<Cupcake> getCupcakes() public String getBottoms(int id) public void changeUserBalance(int totalprice, String username) public int getPriceTop(int id) public int getPriceBot(int id) public String getTops(int id) public User getUser(String username) public void createUser(String username, String password) public void sendOrderToDB(int CUPCAKETOP_ID, int CUPCAKEBOT_ID, int AMOUNT) public boolean comparePassword(String username, String password)