



Cupcake - Olsker Cupcakes

Deltagere:

- Peter E. Marcher
 - Email: cph-pm260@cphbusiness.dk
 - GitHub: Pete3991
- Nikita C. B. B. Nielsen
 - Email: Cph-nn233cphbusiness.dk
 - GitHub: niki03125
- Nicolai Jahncke Strand
 - Email: Cph-ns297@cphbusiness.dk
 - GitHub: Nico538e

Klasse:

- Klasse B

Målgruppe:

- fagfælle på 2. semester

Tidspunkt:

Projektet og rapporten blev udarbejdet i foråret 2025.

Kort introduktion til projektet:

Dette gruppeprojekt handler om vi skal lave full-stack development, med en cupcake shop. Dette betyder at vi både skal lave Frontend og Backend, samt sammenkoble dem med mappers og controllers. Derudover har vi arbejdet med routes, diagrammer og kanban-board, som vi senere vil komme ind på.

Indholdsfortegnelse

01	Indledning/Forside	
	Deltagere	1
	Tidspunkt	1
	Målgruppebeskrivelse	1
	kort introduktion til projektet	1
02	Indholdsfortegnelse	
	indhold	2-3
03	Baggrund	
	Om Olsker Cupcakes	4
	Kunderns mock-up til systemet	4
04	Teknologivalg	
	Oversigt over teknologier	5
	Versionsnumre på anvendt software	5
05	Krav	
	User stories	6
	Firmaets vision og forretningsværdi	6
06	Modeller og diagram	
	Domænemodel	7
	Klasse og Er-diagram med relationer	7
	3. Normalform	8
	Diagram over websider og navigation	8

07	Særlige forhold	
	Exception handling	9
	Sikkerheds foranstaltninger	9
	Valid bruger input	9
08	Status på implementation	
	Hvilke funktioner er færdige	10
	Manglende elementer eller kendte fejl	10
	I sidste øjeblik	10
09	Processen	
	Planlagt arbejdsproces vs. Faktisk arbejdsproces	11
	Godt og dårligt	11
	Forandringer til næste gang	11
	Links	11

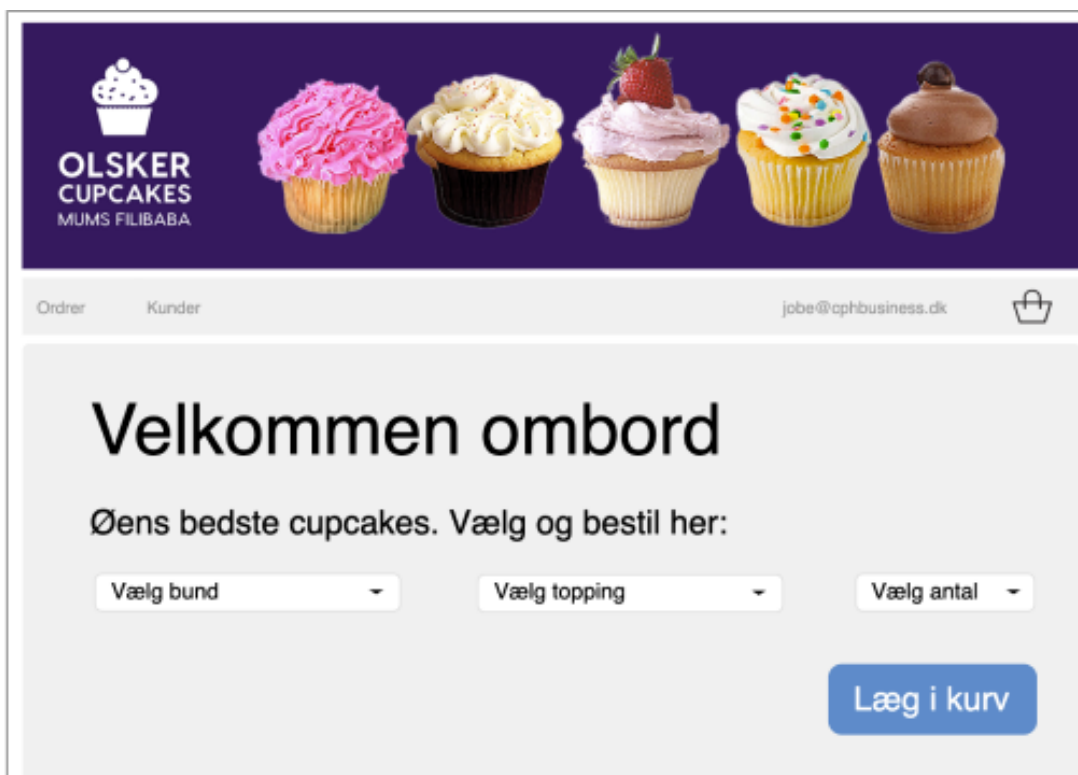
Baggrund

Om Olsker Cupcakes

Olsker Cupcakes er et dybdeøkologisk iværksættereventyr fra Bornholm. Efter besøg fra et par hipstere fra København, har de indsamlet krav og lavet en halvfærdig mockup af en hjemmeside. Vores opgave er at udvikle denne prototype til en fungerende platform. Denne forside viser vores rejse i at forstå kravene, udforske manglende funktionaliteter og give kreative forslag til, hvordan vi kan forbedre løsningen.

Vores tilgang er at tage små etaper ad gangen – ikke lade os forblænde af hipsterfarver, men sørge for en funktionel og brugervenlig oplevelse.

Kunderns mock-up til systemet:



The mockup shows a website for 'OLSKER CUPCAKES MUMS FILIBABA'. The header features a logo and six different cupcakes. Below the header is a navigation bar with 'Ordre' and 'Kunder' links, a user email 'jobe@cphbusiness.dk', and a shopping cart icon. The main content area has a large heading 'Velkommen ombord' and a subheading 'Øens bedste cupcakes. Vælg og bestil her:'. Below this are three dropdown menus: 'Vælg bund', 'Vælg topping', and 'Vælg antal'. A blue button labeled 'Læg i kurv' is positioned to the right of the dropdowns.

Teknologivalg

Oversigt over teknologier:

Projektet er udviklet ved hjælp af en række teknologier for at sikre en stabil og effektiv applikation. Applikationen er bygget i Java og benytter Javalin som webframework til at håndtere HTTP-requests og servere indhold. Til at generere dynamiske HTML-sider anvendes Thymeleaf.

For at gemme og administrere data benyttes PostgreSQL som database, og forbindelsen mellem applikationen og databasen håndteres via JDBC. Byggesystemet Maven anvendes til at styre afhængigheder og projektstruktur, hvilket gør det nemt at vedligeholde og udvide applikationen.

Versionsnumre på anvendt software

1. IntelliJ IDEA 2023.2.5 (Ultimate Edition)
2. Javalin (v6.5.0)
3. Thymeleaf (v3.1.3.RELEASE)
4. SLF4J (v2.0.17)
5. Jackson (v2.18.3)
6. HikariCP (v6.2.1)
7. PostgreSQL JDBC Driver (v42.7.5)
8. Maven Shade Plugin (v3.4.1)

Krav

Firmaets vision og forretningsværdi:

Kundens vision for denne opgave var, at deres brugere (deres kunder) skulle kunne oprette en konto og logge ind på hjemmesiden. Når en bruger var logget ind, skulle de kunne vælge frit mellem de forskellige muligheder for bunde og toppe samt vælge det ønskede antal cupcakes. Herefter skulle prisen for bestillingen beregnes og vises i en indkøbskurv, så brugeren kunne se den samlede pris, før ordren blev bekræftet. Betalingen skulle trækkes fra brugerens konto, hvor saldoen blev administreret i en database.

En administrator skulle kunne indsætte beløb på en kundes konto direkte i systemet. Derudover skulle administratoren have adgang til at se alle ordrer, der var blevet afgivet.

Når en bruger eller administrator var logget ind, skulle deres e-mail vises på hver side.

User Stories

US-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.

US-2:

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

US-3:

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

US-4:

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

US-5:

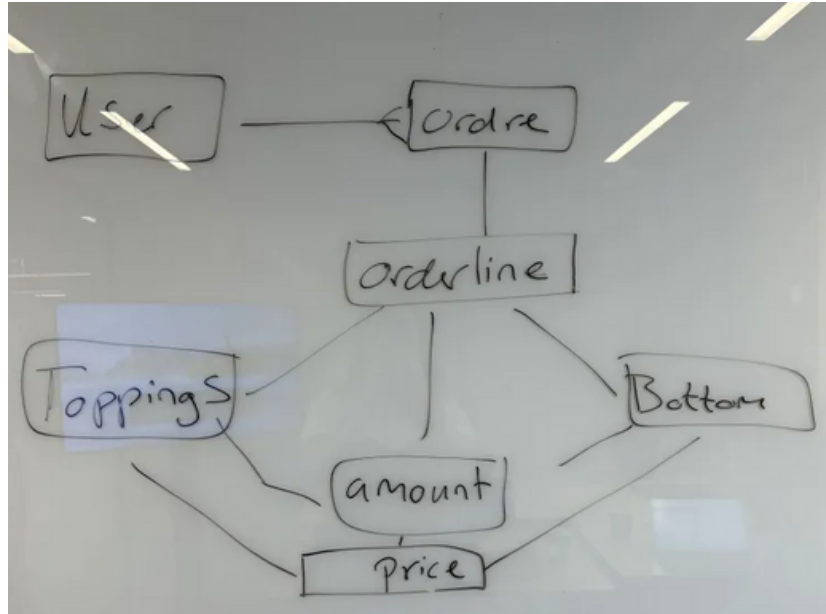
Som kunde eller administrator kan jeg logge på systemet med email og kodeord. Når jeg er logget på, skal jeg kunne se min email på hver side (evt. i topmenuen, som vist på mockup'en).

US-6:

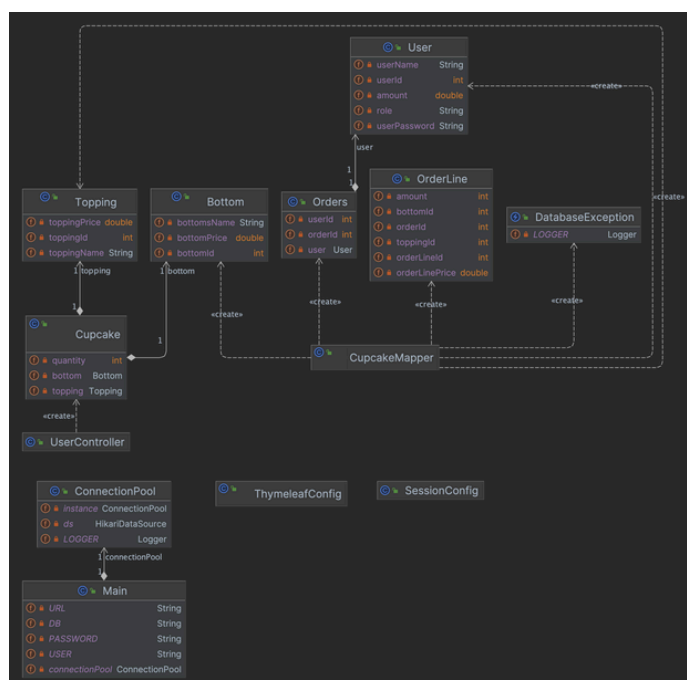
Som administrator kan jeg se alle ordrer i systemet, så jeg kan se hvad der er blevet bestilt.

Modeller og diagram

Domænemodel:



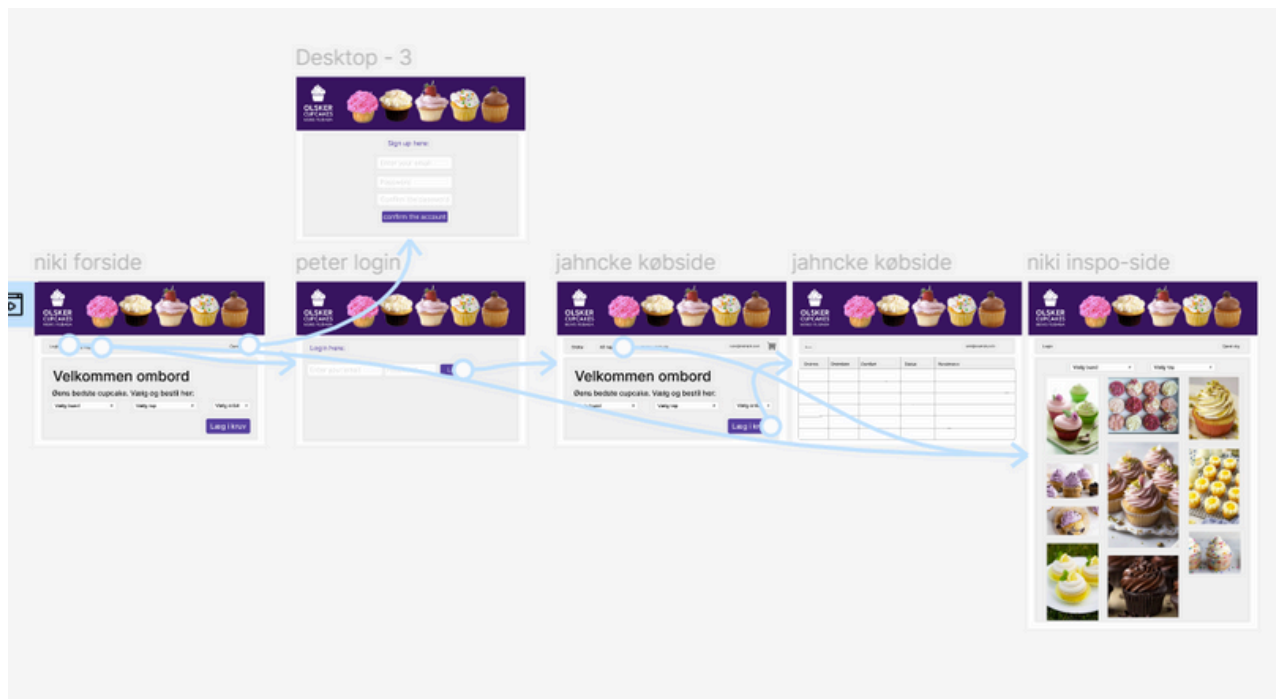
Klasse og Er-diagram med relationer:



3. Normalform:

Vi har bygget vores database med fokus på at holde den overskuelig og nem at arbejde med. Derfor har vi tænkt over, hvordan vi kunne undgå gentagelser og placere oplysninger dér, hvor de giver bedst mening. Det kaldes at følge 3. normalform, og det hjælper med at sikre, at data ikke modsiger hinanden, og at vi undgår forvirring. På den måde bliver databasen nemmere at holde styr på – både når man bygger systemet og senere skal rette eller udvide det. Og selvfølgelig når vi overholder 3NF så overholder vi også 1 og 2NF.

Diagram over websider og navigation:



Særlige forhold

Exception handling:

Igennem denne klasse bliver der håndteret database Exception errors, som der bruges til at fange fejl. Hvis der er noget i java koden som ikke stemmer overens med dataene i databasen skal det fanges og der skal sendes en fejlbesked. Derudover skal der eventuelt gøres noget funktionelt så man forbliver på den samme side eller andet.

```
public class DatabaseException extends Exception { 43 usages ▲ Nico538e +1
{
    private static final Logger LOGGER = Logger.getLogger(DatabaseException.class.getName()); 3 us
    public DatabaseException(String userMessage) { 4 usages ▲ Nico538e
    {
        super(userMessage);
        LOGGER.log(Level.WARNING, userMessage);
    }

    public DatabaseException(String userMessage, String systemMessage) { 3 usages ▲ Nico538e
    {
        super(userMessage);
        LOGGER.log(Level.WARNING, userMessage);
        LOGGER.log(Level.WARNING, msg: "errorMessage: " + systemMessage);
    }

    public DatabaseException(String userMessage, SQLException e) { 10 usages ▲ Peter399c1
    {
    }
}
```

Sikkerheds foranstaltninger:

Sikkerheds foranstaltningerne i vores projekt er de grænser der bliver sat, for hvad brugeren skal kunne gøre og kunne se. En bruger skal f.eks. ikke kunne gå til administrator sider og se andre brugers oplysninger og det kun admin brugere der skal blive sendt til de sider. En bruger skal kunne oprette sig og logge ind og bestille og købe et produkt og der sættes der nogle grænser for hvad en bruger skal kunne gøre funktionelt på Hjemmesiden.

Valid bruger input:

Brugerens input skal kunne valideres når den potentielle bruger skal oprette sig, så skal deres E-mail og password valideres og der er sat nogle grænser op for hvad brugeren kan.

Hvis E-mailen allerede er gemt i databasen så betyder det at brugeren allerede er eksisterende og man kan derfor ikke oprette en ny bruger med de samme tegn. Derudover skal man når man opretter sig skrive sit password ind og derefter bekræfte det ved at skrive det igen.

Men hvis det ikke er det samme password så kan brugeren ikke oprettes.

Brugeren skal prøve igen på samme side, på den måde dobbelt tjekker vi passwordet og sikrer at brugeren har skrevet det rette password de vil bruge.

Status på implementation

Hvilke funktioner er færdige:

I vores projekt har vi fået implementeret user stories 1-5. En bruger kan logge ind eller oprette sig som en ny bruger, hvor den gemmer brugers data i en database.

Brugeren kan vælge mellem de bunde og toppe som vi også har liggende i databasen og bestille en cupcake. Brugeren kan til sidst så betale med den saldo de har liggende.

Hvis man logger ind som admin kommer man hen til en anden side som viser alle kunderne af butikken.

Manglende elementer eller kendte fejl:

På forsiden sker der en server fejl hvis man ikke har valgt en bund eller top til sin cupcake og trykker læg i kurv. Vi mangler måske en "gå tilbage" knap, hvis brugeren ikke ved at man kan trykke på logoet for at komme tilbage til forsiden.

På user story 6 mangler vi at kunne vise alle ordrer i systemet så man kan se hvad der er blevet bestilt når man er logget ind som administrator.

Vi mangler en fejlmeddelse til når man prøver at betale for cupcakes, men ikke har flere penge på saldoen.

I sidste øjeblik :

Der har været lidt ting som er sidste øjeblik's ting, så som at få det sidste på plads med US 6. US 6, handler om at man som administrator skal kunne se hvert enkelte brugers ordrer

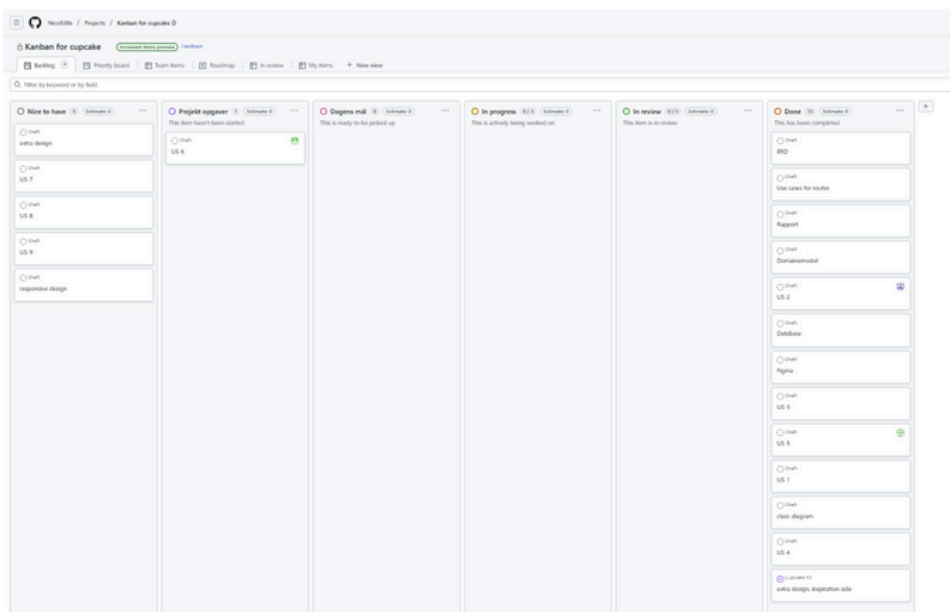
I US 6, kort fortalt blev vi færdige med en mapper metode og en html side men sammenkoblingen mellem dem er ikke helt faldet på plads, vi mangler en metode i controlleren og nogle routes.

Vi har desværre ikke overholdt vores kodeStop som vi skulle, så vi kom lidt for sent i gang med rapporten. Som vi også nævner senere i "forandringer til næste gang".

Processen

Planlagt arbejdsproces vs. Faktisk arbejdsproces:

Vi lavede et kanban-board i starten af vores projekt, for at nemmere kunne holde styr på alle vores opgaver, user stories, need to have og nice to have. Vi har fulgt den nogenlunde men er kommet i mål med næsten alle user stories, det eneste vi mangler er det sidste på US 6. som vi har nævnt tidligere.



Godt og dårligt:

Som gruppe er det anden gang vi arbejder sammen, det har fungeret rigtig godt. Vi har haft en åben kommunikation i gennem hele projektet, hvor vi frit har kunne spare med hinanden og hjælpe andre hvor vi har vores styrker. Vi kunne godt havde tænkt os flere muligheder for vejledning, samt længere tid til mandags vejledningen. Derudover fandt vi desværre hurtigt ud af at det var forskellig standpunkter, omkring hvordan dette project skulle forløbe, som var lettere forvirrende. Vi var også lidt forvirret om hvor vi var i mandags, men det fandt vi heldigvis ud af hurtigt mandag middag.

Forandringer til næste gang:

Næste gang skal vi måske tænke lidt på at lave en ting af gangen, sådan så vi får klaret user stories en efter en, i stedet for at have halvdelen klaret på halvdelen af dem. En tydeligere kodelstop som vi alle skal følge, ville gavne os til næste gang.

Links:

Youtube Link: <https://youtu.be/ScBgIA4eO90>

GitHub: <https://github.com/Nico538e/Cupcake>