Forside

Projekt: Olsker Cupcakes

Deltagere:	Cph-Mail	Git-brugernavn
Rikke Mariussen	cph-rm225@cphbusiness.dk	RikkeMariussen
Casper Gervig	cph-cg201@cphbusiness.dk	Gervig
Rasmus Højholt	cph-rs255@cphbusiness.dk	RHoejholt
Daniel Kroner	cph-dn138@cphbusiness.dk	DanielKroner
Christoffer Leisted	cph-cl446@cphbusiness.dk	Crispy212

Klasse: Datamatiker F24

Tidspunkt: Uge 43/44 2024

Indholdsfortegnelse

Forside **1**
Projekt: Olsker Cupcakes 1
Indholdsfortegnelse **2**
Indledning **3**
Baggrund **4**
Teknologi valg

IDE: IntelliJ IDEA 2023.3.4 ..... **5**
Krav **6**
Firmaets håb og ønsker 6
User-stories 6
Aktivitetsdiagram **7**
Domæne model og ER diagram **8**
Domænemodel 8
ER diagram 10
Navigationsdiagram **12**
Særlige forhold **13**
Status på implementation **14**
Proces **14**
Gik godt: 15
Kunne være bedre: 15
Links **16**

Indledning

Denne rapport er tiltænkt studerende på andet semester på datamatikeruddannelsen. Rapporten vedrører et projekt præsenteret for os af en virksomhed som sælger cupcakes på Bornholm. Virksomheden ønsker en hjemmeside, hvor man som kunde kan bestille cupcakes til afhentning i deres butik via en profil som kunden selv opretter. Dertil skal virksomhedens ansatte kunne tilgå hjemmesiden som administratorer, og kunne se samt tilrette de forskellige brugere og deres ordrer, samt deres saldo. Endvidere gav virksomheden et mock-up af hvordan de ønsker hjemmesiden ser ud, user-stories om de funktionelle krav, og en liste af ikke-funktionelle krav.

Denne rapport er baseret på denne præsentation og oplysninger fra virksomheden. Der er videreudviklet på mock-up af designet, oprettet en database til de oplysninger som virksomheden skal have gemt i forbindelse med ordre, bestillinger og kunderne som er nødvendige til hjemmesidens funktionalitet, samt en påbegyndelse af full stack udviklingen af hjemmesiden.

Baggrund

Olsker Cupcakes er et af Bornholms iværksættereventyr, som ønsker at samarbejde med os. Projektet fra dem består i at lave en hjemmeside til dem på baggrund af et mock-up, med et logo. Hjemmesiden skulle gøre det muligt for forbrugere at oprette en profil hos dem, og derefter kunne bestille cupcakes og betale via hjemmesiden, for derefter blot at afhente dem i butikken i Olsker.
Ved at bestille via hjemmesiden kan forbrugeren selv sammensætte hvilke bunde og toppings de vil have, ud fra lister med tilgængelige smage. Endvidere vil forbrugeren på hjemmesiden kunne se deres tidligere ordrer og bestillinger.

På administrationssiden skulle de ansatte kunne se de oprettede konti for deres forbrugere, deres ordre og have muligheden for at tilføje et beløb til individuelle konti efter behov. Derudover skal de også kunne se alle ordrer lavet på deres hjemmeside.

Alle brugere, både kunder og administratorer samt ordre og de forskellige cupcakes vil blive gemt i en database.

Alle disse krav er kommet i form af User Stories, som er blevet brugt til udvikling af produktet til kundens efterspørgsel.

Teknologi valg

IDE: IntelliJ IDEA 2023.3.4

Containerization: Docker Desktop 4.34.3
Database Management: PostgreSQL 42.7.2, PGAdmin 4 (version 8.3)
JDK: Amazon Corretto 17 (version 17.0.12)
Framework: Javalin 6.1.3
Template Engine: Thymeleaf 3.1.2 (Release), Thymeleaf-extras 3.0.4 (Release)
Frontend: CSS3, HTML5
Build Tool: Maven (compiler version 3.13.0)
Encoding: UTF-8
JDBC Driver: JDBC 4.2
Interface Design Tool: Figma Desktop App Version 124.5.5

Krav

Firmaets håb og ønsker

Firmaet ønsker at skære ned på tidsforbrug vedrørende ordrebestilling og samtidig kunne tilbyde deres kunder en nemmere og hurtigere måde at bestille deres cupcakes på.
Før skulle de (evt.) bruge en medarbejder til at tage imod ordrer telefonisk og fysisk i butikken. Nu behøver de ingen medarbejdere til fjern-ordrer, da deres kunder selv kan bestille via deres hjemmeside. Desuden har deres kunder nu også en nemmere måde at undgå ventetid på, både for at komme i kontakt med butikken, men forhåbentligt også som walk-in kunder. Medarbejdere kan nu i højere grad fokusere på at betjene kunder i butikken, omend det er bestilling eller vareudlevering.

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).

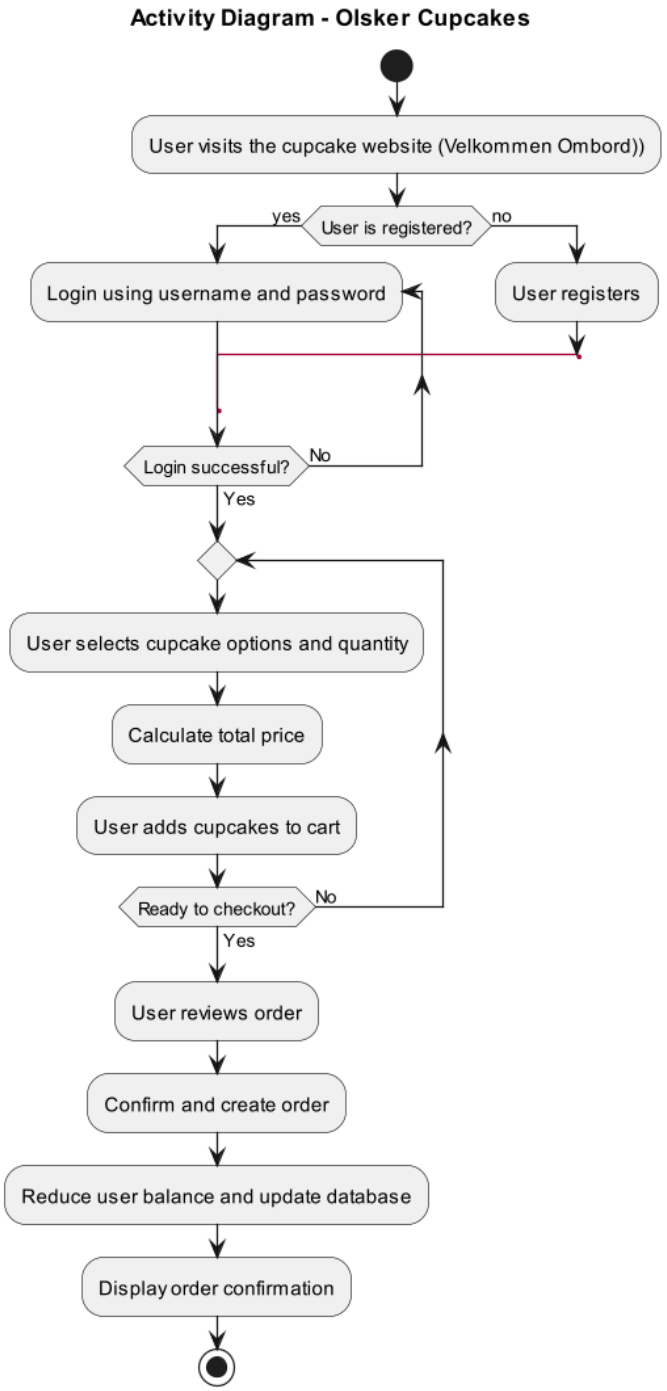
US-6: Som administrator kan jeg se alle ordrer i systemet, så jeg kan se hvad der er blevet bestilt.

US-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.

US-8: Som kunde kan jeg fjerne en ordrelinje fra min indkøbskurv, så jeg kan justere min ordre.

US-9: Som administrator kan jeg fjerne en ordre, så systemet ikke kommer til at indeholde ugyldige ordrer. F.eks. hvis kunden aldrig har betalt.

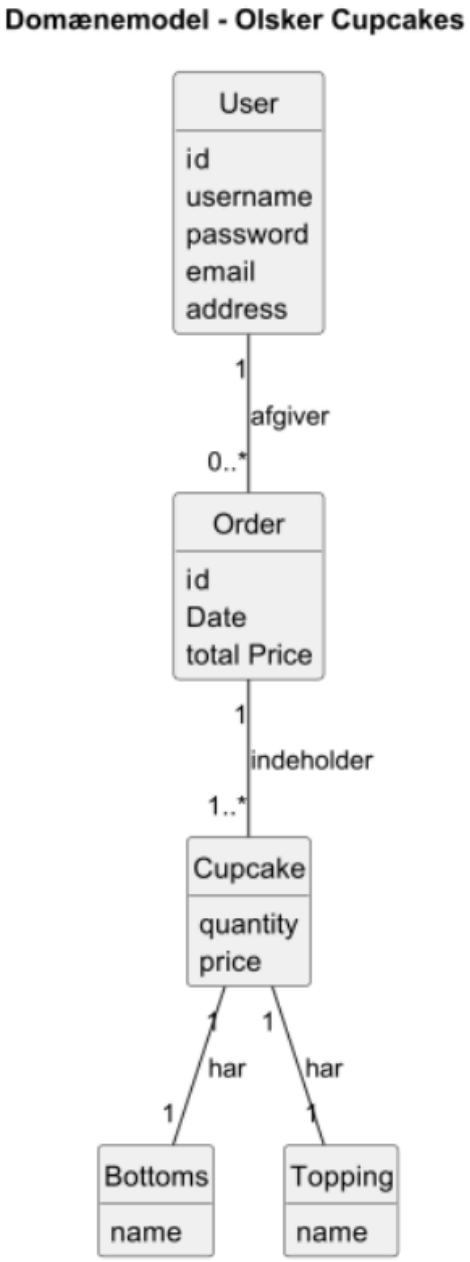
Aktivitetsdiagram



Aktivitetsdiagrammet viser den overordnede gennemgang af vores webside. Her kan man se hvordan man kommer frem til forskellige funktioner.

Domæne model og ER diagram

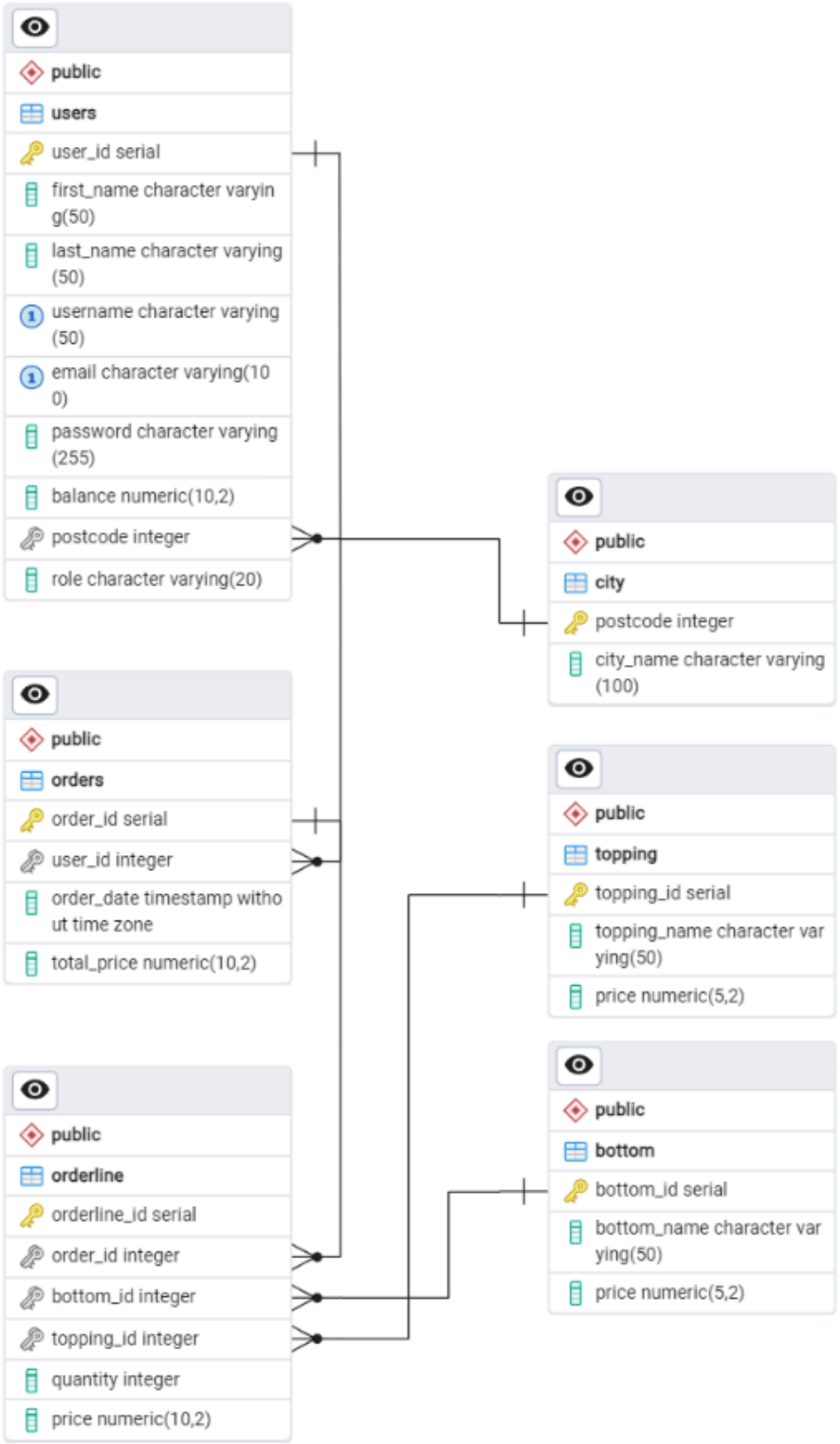
Domænemodel



En User kan afgive nul eller flere ordrer, og hver ordre bliver afgivet af præcis én user.
En Order indeholder en eller flere cupcakes hver cupcake er en del af præcis én order.
Hver Cupcake har præcis én Bottom, og hver Bottom kan bruges af en eller flere cupcakes.
Hver Cupcake har præcis én topping og hver topping kan bruges af en eller flere cupcakes.

Relationen mellem en cupcake og dens bottom og topping kan udvides til at tillade, at en cupcake har flere toppings for mere komplekse kombinationer, men dette ville komplicere modellen og øge implementerings kompleksiteten.

ER diagram


public
users
user_id serial
first_name character varyin g(50)
last_name character varying (50)
username character varying (50)
email character varying(10 0)
password character varying (255)
balance numeric(10,2)
postcode integer
role character varying(20)
public
city
postcode integer
city_name character varying (100)
public
orders
order_id serial
user_id integer
order_date timestamp witho ut time zone
total_price numeric(10,2)
public
topping
topping_id serial
topping_name character var ying(50)
price numeric(5,2)
public
orderline
orderline_id serial
order_id integer
bottom_id integer
topping_id integer
quantity integer
price numeric(10,2)
public
bottom
bottom_id serial
bottom_name character var ying(50)
price numeric(5,2)

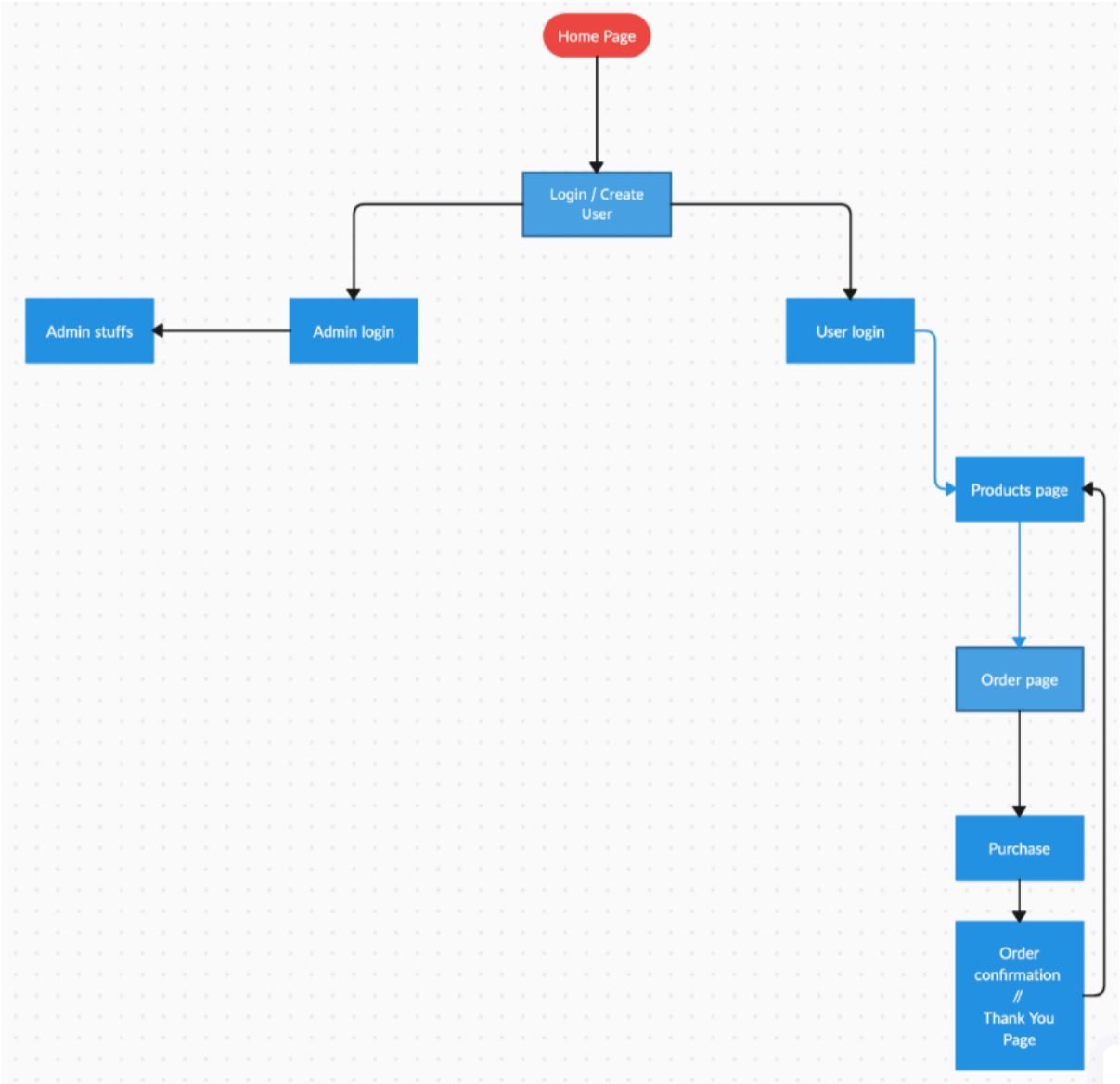
Databasen opfylder de tre normalformer:

1. Normalform: Tabellen har atomare værdier, hvilket betyder, at hver kolonne kun indeholder én værdi per række, og der er ingen gentagne grupper. Alle kolonner indeholder enkeltværdier (f.eks. en balance, en dato osv.).

2. Normalform: Databasen har ingen delvise afhængigheder, da alle ikke-nøgleattributter er fuldt funktionelt afhængige af den primære nøgle. For eksempel afhænger attributter i orderline (som quantity og price) fuldt ud af den sammensatte nøgle mellem order_id og bottom_id/topping_id.

3. Normalform: Databasen har ingen transitiv afhængighed. Alle ikke-nøgleattributter er afhængige af kun primærnøglerne i tabellerne. Eksempelvis i users tabellen afhænger attributter som balance, email, osv., kun af user_id, og der er ingen transitiv afhængighed til andre attributter.

Navigationsdiagram



Man kan ikke tilgå sortimentet uden at logge ind, man kan heller ikke placere en ordre. Vi kræver at man logger ind for at placere ordren, da vi skal vide hvem der gør det og om de kan betale det de bestiller.

Der er desuden et admin login, som har mulighed for at tilgå administrative funktioner, det inkluderer at kunne se kunder og alle ordrer i systemet.

Særlige forhold

Man kan oprette og logge ind som bruger, så brugeren bliver gemt i en session. Kundens ordre vil blive gemt i en indkøbskurv, som man kan få vist. Den bliver også tildelt et ordrenummer, men man kan ikke bestille selve ordren.

I vores mapper klasser har vi metoder som eksekverer en SQL query i en try catch, som bliver sendt til vores database. Hvis den query er forkert vil metoden give en SQL exception som fanges af catch. Vores catch kaster en DatabaseException som vi selv har lavet, hvor der skrives ud i konsollen hvor og hvad gik galt. Det er brugbart når man skal debugge systemet.

I vores html har vi f.eks en input type som hedder email, så det skal indeholde et @, dette er en feature der blev implementeret med HTML5. Når man opretter bruger skal man skrive sit kodeord 2 gange, der valideres at kodeordene som skrives er ens. Brugerne skal heller ikke kunne oprette en bruger med samme e-mail eller brugernavn som en allerede eksisterende bruger. Man kan heller ikke indskrive et postnummer, som ikke findes i vores database.
For alle vores input bruges der HTML required funktionen, så brugeren skulle skrive et input i de felter der er opstillet.

Når man logger ind, så findes der 2 roller i databasen, admin eller customer. Brugeren bliver automatisk oprettet med customer rollen hvis brugeren oprettes via websiden. Den eneste måde man kan blive administrator er ved at oprette en bruger direkte i pgAdmin. Denne rolle er gemt som en “character varying” i databasen og behandlet som en String i vores backend.

Status på implementation

Vi har udviklet de fleste websider som illustreret i vores navigationsdiagram, men de er endnu ikke helt færdigudviklede.
Det er muligt at tilføje cupcakes til indkøbskurven, men vi mangler funktionen til at gennemføre køb, hvor pengene trækkes fra kundernes balance. Vi mangler også muligheden for, at administratorer kan slette ordrer og indsætte penge på kunders balance. Der er også steder i vores backend som sender en videre til en error.html side, men sådan en side har vi ikke.

Vores side til indkøbskurven mangler noget styling, ellers er de andre sider færdig stylet.
Vi opdagede også i sidste øjeblik, at nye oprettede kunder ikke kan bestille cupcakes.
Vores administrator-sider kan tilgås ved at skrive HTML-stien direkte i URL'en, hvilket betyder, at der mangler et ekstra sikkerhedslag for at forhindre uautoriseret adgang.
Vi har sat en test mappe op med en IntegrationTest klasse, vi har fået testet en enkelt metode. Men det meste af det arbejdet med tests er at få det sat, så vi skulle bare have kaldt vores metoder, vi ville have testet.

Der er også små fejl og mangler, som er blevet opdaget i brugertest. Der mangler en pris på de individuelle bottoms og toppings, når man er inde for at bestille dem (ikke på ordrelisten). Bruger synes også det var irriterende, at laver man fejl i (password) brugeroprettelse, bliver alle ens indtastninger nulstillet. Email-adressen på den aktive bruger bliver heller ikke vist oppe i vores navigationsbar.

Vi har også en totalprice kolonne i vores orders tabel som skulle kunne udregne den totale pris for ordren, men den funktion har vi ikke fået implementeret. Den viser altid 0 nu.

Proces

Vores plan var at bruge Agile metoden Kanban til at opdele arbejdet. På vores kanban board delte vi opgaverne op i små bidder og vi kunne se hvad alle lavede og hvor langt de var nået.
Vi lavede også en code of conduct for hvordan vi bruger git og GitHub.
Her er vores code of conduct:

Start i dev branch Skift til dev branch, hvis ikke du allerede er der.

Opdater projektet Synkroniser projektet ved at vælge "Update Project" (Ctrl+T / CMD+T eller via menuen GIT -> Update Project), så du henter de nyeste ændringer.

Lav en ny feature branch Opret en ny branch ved at højreklikke på dev branch og vælge "New Branch" → Navngiv din feature branch.

Arbejd i din feature branch Implementér dine ændringer i den nye branch.

Commit ofte og brug gode commit-beskeder Lav hyppige commits med meningsfulde beskeder. Klik på "Commit" i IntelliJ, og skriv en klar og beskrivende besked for hver ændring.

Før push, opdater dev branch Skift til dev branch, og brug "Update Project" igen for at hente de nyeste ændringer.

Gå tilbage til din feature branch og merge dev ind Skift tilbage til din feature branch og merge dev branch ved at trykke på drop down, vælge dev og klikke "Merge Dev into *Your branch name*". Løs eventuelle konflikter.

Test om alt virker Test, at din kode fungerer som forventet efter sammensmeltningen.

Lav en pull request fra din feature branch til dev branch Opret en pull request (PR) direkte fra IntelliJ eller via GitHub, afhængigt af jeres præference.

Praksis

Det gik godt med at få det sat op, men at opdatere vores kanban board var ikke helt nemt. Så det var ikke til at vide hvilke opgaver der var færdige og hvad folk lavede.
Vi prøvede også at holde os til vores code of conduct med hensyn til git og GitHub, men det kunne godt drille, så det var ikke altid lige til.

Gik godt

Vi var gode til at sætte kanban boardet op og få fordelt arbejdet, så vi kunne komme i gang. Vi kunne hjælpe hinanden hvis vi sad fast. Vi var gode til at tage pauser som gruppe, så vi ikke gik helt død. Når vi arbejdede, så gjorde vi det som en gruppe, så vi ikke enkeltvis sad og knoklede. Vi var gode til at arbejde i vores egne feature branches, så vi ikke sad og skrev i den samme branch.

Kunne være bedre

Vores planlægning og kommunikationen kunne være bedre. Vores hjemmeside er ikke helt færdig og opfylder ikke alle vores usecases. Vi kunne være bedre til at bruge pull requests til vores repository. Vores kanban board blev ikke opdateret nok, så det var svært at få et overblik om hvad vi manglede og hvad andre lavede senere i projektet. Vi var heller ikke gode til at lave alle diagrammerne på forhånd, eller til at tage dem i brug. Vi skal blive bedre til at gøre det som noget af det første, og bruge dem som et roadmap, der løbene bliver itereret på.

Forbedring

Vi skal arbejde på kommunikationen.
Som gruppe skal kommunikationen være bedre mellem medlemmer som møder op og dem som tager en dag hjemme, da vi løb ind i problemer med at få status på forskellige dele af projektet. Dette gjorde at der til tider var stor tvivl om hvor langt i projektet vi egentlig var.

Links

Link til youtube video der demonstrerer funktionaliteten af vores webside: videolink

Link til figma fil: Figma for Olsker Cupcakes