Cupcake Projekt

Gruppemedlemmer

Masih Bijan Kabiri, cph-mk330@cphbusiness.dk, masih1903

Tobias Welling Hansen, cph-th401@cphbusiness.dk, TobiOneCanobi

Peter Janas, cph-pj282@cphbusiness.dk, peterjanas

Henrik Thunbo Jensen, cph-hj286@cphbusiness.dk, HenrikThunbo10

Umair Tafail, cph-ut38@cphbusiness.dk, Umair-230104

Datamatiker 2.Semester, Klasse B

Lavet 2. April 2024 - 11. April 2024

Indholdsfortegnelse

Indledning

Baggrund

Denne rapport beskriver udførelsen af en opgave, vi er blevet stillet af en virksomhed. Formålet med rapporten er at skabe klarhed for en fagfælle, en medstuderende på 2. semester på Datamatiker-uddannelsen, så de vil kunne forstå vores løsning og hvordan vores workflow har været.

Vores team skal lave en hjemmeside for Olsker Cupcakes, som er en virksomhed der sælger cupcakes med forskellige smagsvarianter. Virksomheden er en starter virksomhed med en vision om at skabe de bedst smagende cupcakes med økologiske fødevarer.

En anden gruppe fra København har på forhånd udarbejdet et mock-up, en løs skitse af hvordan den færdige website eventuelt kunne se ud. Vi har ladet os inspirere af deres skitse og har derfor udviklet hjemmesidens forskellige sider ud fra den. Vi har dog selv udarbejdet et overblik over resten af projektets indhold ved brug af diagrammer, kanban-board og user stories.

Teknologivalg

IntelliJ IDEA 2023.2.2
Vi har udarbejdet vores løsning til kunden i programmeringssproget Java, et sprog der i branchen er velkendt og anvendes mange steder. Vi har skrevet vores kode i en software ved navn IntelliJ.

JDBC
Java Database Connectivity er et framework i kodning, som man kan implementere med et givent programmeringssprog. Frameworket tillader en programmør at skabe forbindelse til en database fra en kodning software som IntelliJ, og derved redigere databasens indhold gennem interaktion på en hjemmeside som både bruger og administrator.

PostgreSQL
Denne hjemmeside er et system hvor man kan skabe og redigere databaser med SQL sproget.

Javelin
Et open-source web framework for Java

Thymeleaf
Et redskab der kan bruges til at forbinde java-metoder med databaseindhold til html kode.

Krav

Firmaets håb og vision for systemet

Firmaets håb med dette system er at optimere virksomhedens forretningsproces og øge deres salg af cupcakes. Virksomhedens vision er at skabe en effektiv og brugervenlig måde for kunderne at bestille og købe cupcakes på. Ved at implementere vores program kan virksomheden få en udvidet kundebase, bedre dataindsigt samt levere en bedre kundeoplevelse.

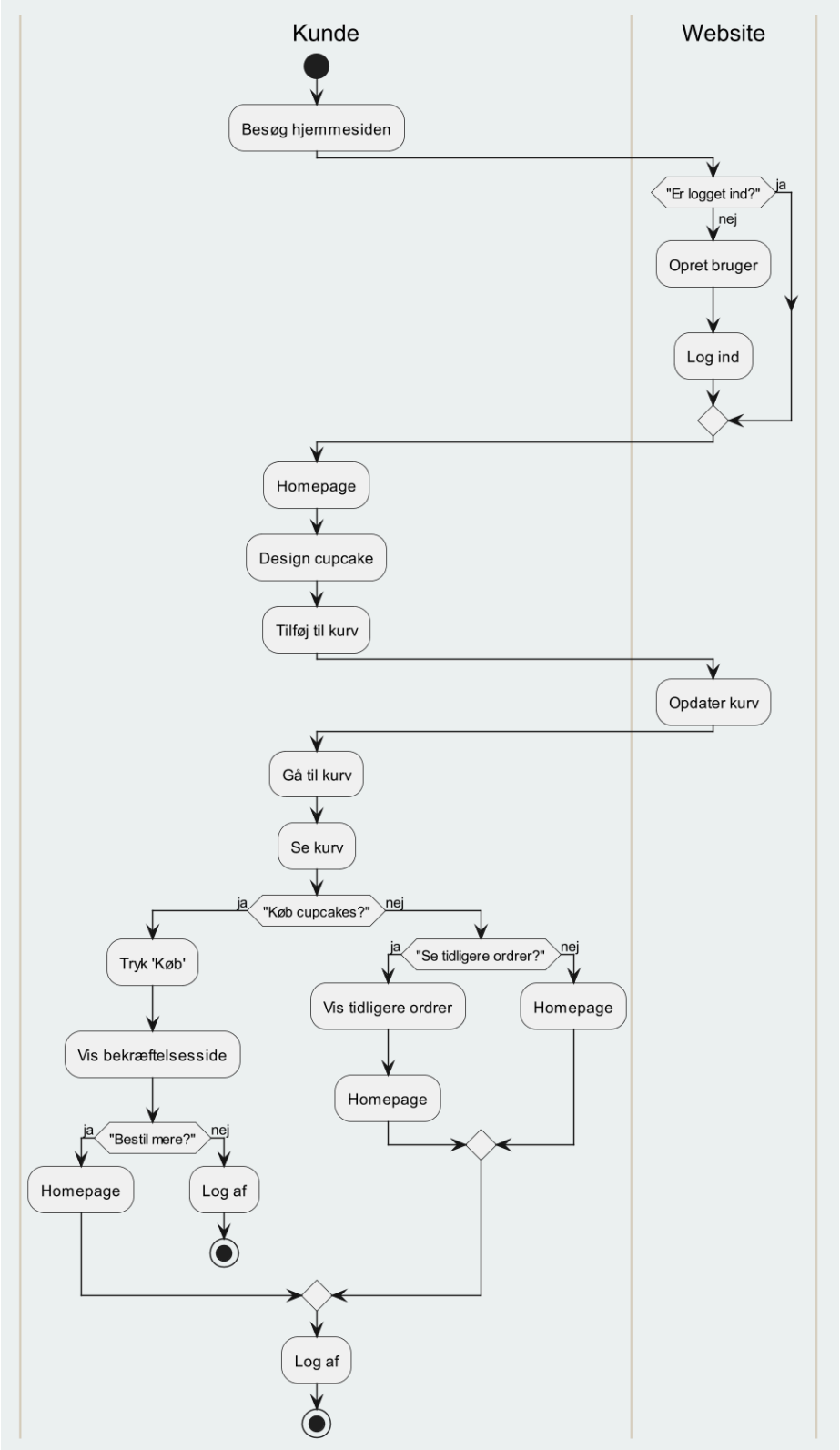
User stories

User stories	**Status**
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.	**Tjek**
US-2 Som kunde kan jeg oprette en konto/profil for at kunne betale og gemme en ordre.	**Tjek**
US-3: Som administrator kan jeg indsætte beløb på en kundes konto direkte i Postgres, så en kunde kan betale for sine ordrer.	**Tjek**
US-4: Som kunde kan jeg se mine valgte ordrelinjer i en indkøbskurv, så jeg kan se den samlede pris.	**Tjek**
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 mock up'en).	**Tjek**

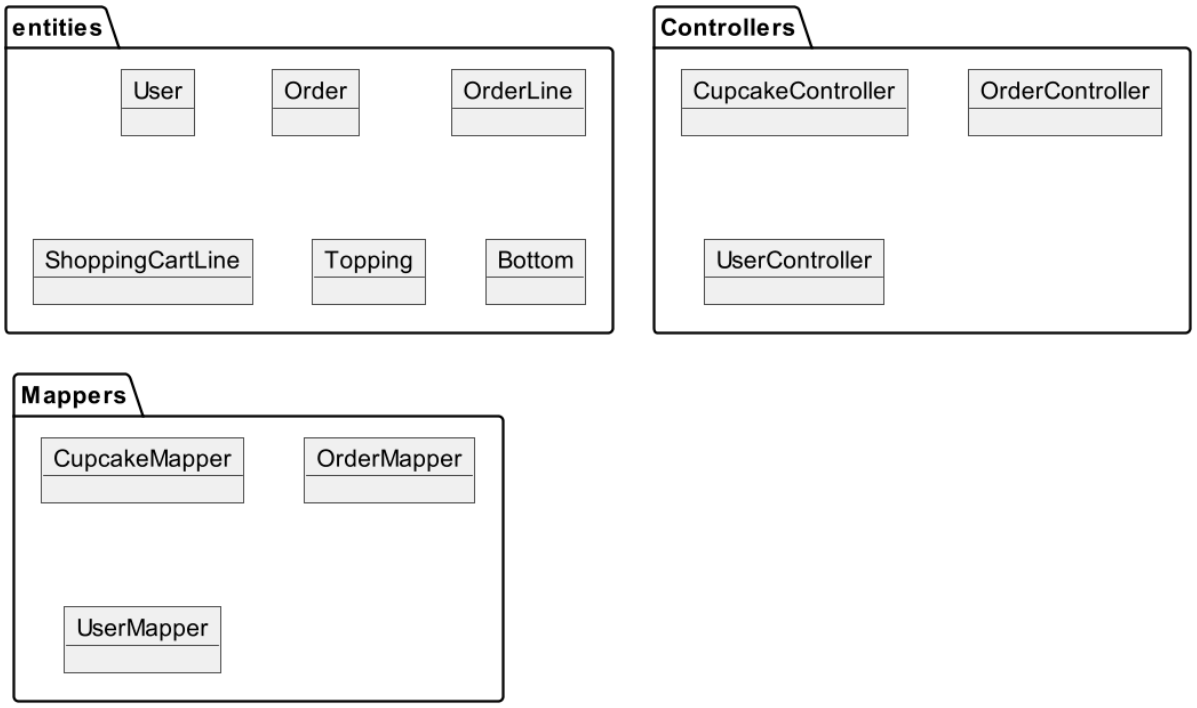
US-6: Som administrator kan jeg se alle ordrer i systemet, så jeg kan se hvad der er blevet bestilt.	**Tjek**
US-7: Som administrator kan jeg se alle kunder i systemet og deres ordrer, sådan at jeg kan følge op på ordrer og holde styr på mine kunder.	**Tjek**
US-8: Som kunde kan jeg fjerne en ordrelinje fra min indkøbskurv, så jeg kan justere min ordre.	**Mangler**
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.	**Mangler**

Diagrammer

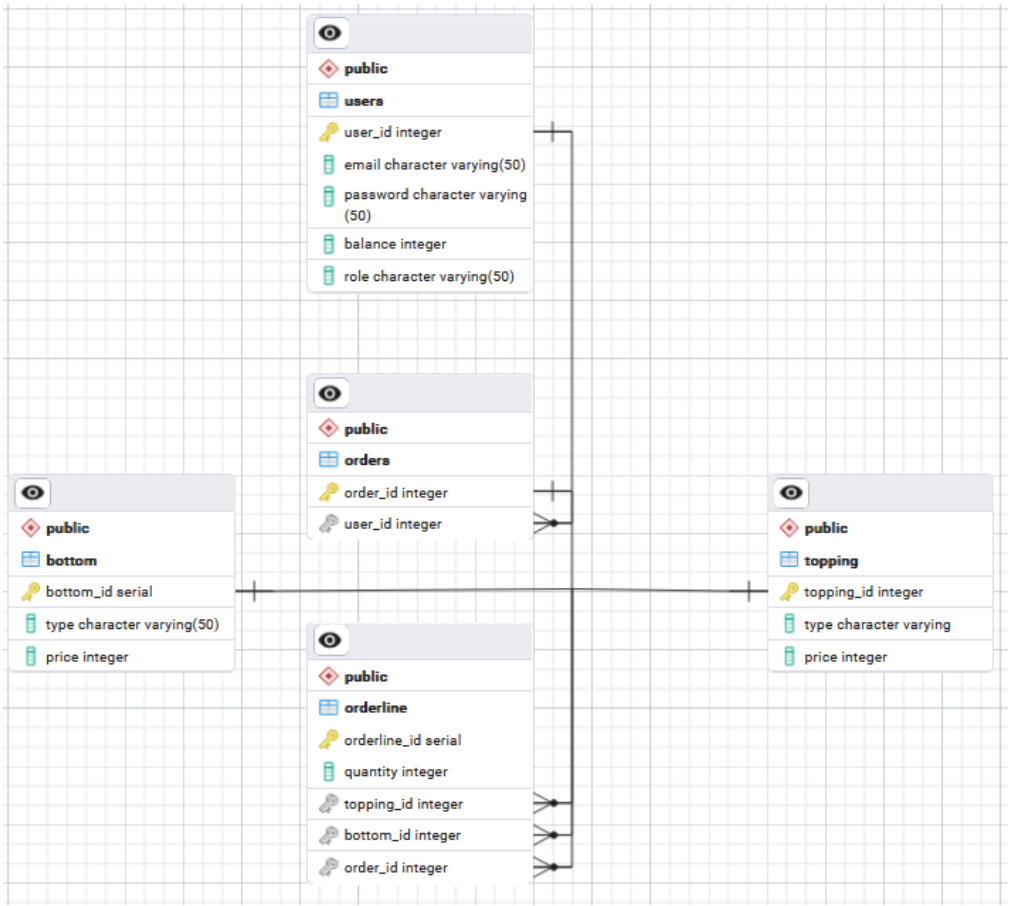
Aktivitetsdiagram



Domæne model

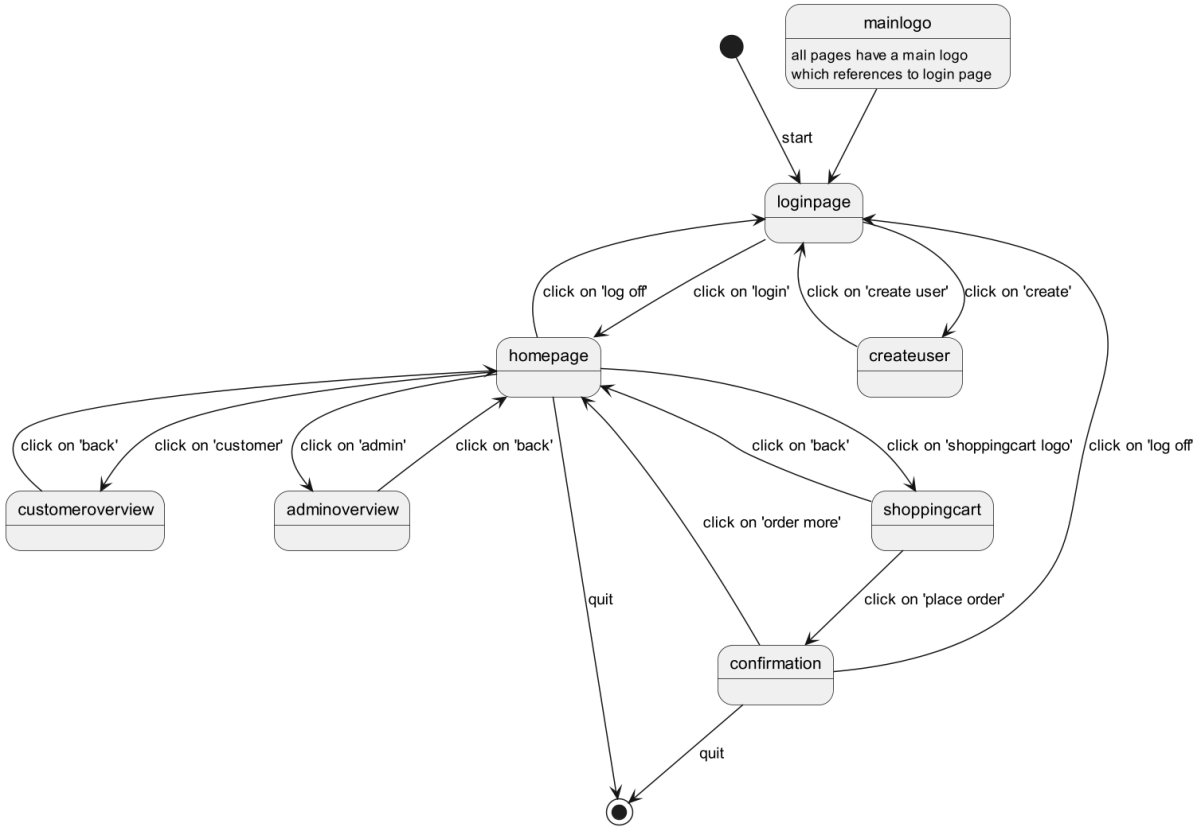


ER diagram



Vores ER-diagram omfatter 5 tabeller, der er relationelt forbundet, og de opfylder alle tre normalformer. Alle tabeller har en-til-mange-relationer, med undtagelse af bottom- og topping-tabellen. De to tabeller har en-til-en-relation og vi har valgt at ikke sammensætte det til en tabel. Dette er fordi vores program kun omhandler salget af cupcakes. Hvis programmet havde været større og omfattet flere typer af produkter, ville det være hensigtsmæssigt at oprette en tabel. Fremmednøgler forbinder vores tabeller i et hierarki, hvilket muliggør adgang til en retning.

Navigationsdiagram



Særlige forhold

Dette afsnit bruges til at beskrive særlige forhold der benyttes i programmet. Det kan f.eks. være:

- Hvilke informationer der gemmes i session

currentUser samt current Users e-mail bliver gemt som en session attribute, brugerens shopping cart bliver også gemt i sessionen.

- Hvordan vi håndterer man exceptions

Vi har valgt at lave en pakke til kun exceptions som håndterer alle exceptions på samme måde

- Hvordan vi har valgt at lave validering af brugerinput

Der bliver lavet validering af brugerinput når man forsøger at logge ind, her vil der selvfølgelig blive checket om ens input er i databasen før man kan komme videre. Ved oprettelse af en ny bruger bliver der checket om ens e-mail allerede bliver brugt, idet en e-mail skal være unik. Desuden skal en gyldig e-mail indeholde et '@' tegn, og et password kun må indeholde standard bogstaver og tal og skal minimum være 4 tegn lang.

- Hvordan vi har valgt at lave sikkerhed i forbindelse med login

En bruger kan have rollerne customer eller admin. Admins har adgang til vores admin overview side, hvor man kan se alle ordrer og hvilke brugere der har bestilt dem. Det kan en normal kunde ikke, så der har vi et check på vores html side om, at der skal vises admin overview knappen eller ej. Vi har også brugt PreparedStatement i vores mappers i stedet for Statement for at undgå en SQL-injection., så en bruger eller ny besøgende ikke kan ændre databasen ved en fejl

Implementeringsstatus

Vi har nået at lave alle de websider der er planlagt for vores færdige produkt.

Vi har næsten nået at lave alle CRUD (create, read, update, delete) metoder, for at de forskellige brugere af hjemmesiden har flere muligheder for at behandle deres ordrer. Vi mangler muligheden for at en bruger skal kunne slette elementer af sin shopping cart og at en admin via vores admin overview page, skal kunne slette ordrer fra vores database.

Vi har fået stiliseret alle vores sider ud efter vores indledende design. Indtil videre har vi ikke fået centreret nogle få af vores elementer automatisk, men i stedet hard-coded. Vi har heller ikke udført scalability, så siden tilpasser sig andre platforme end pc.

En feature vi prøvede at lave som kun blev gjort halvfærdig er at en bruger der er logget ind og som kommer tilbage til login side, skal ikke kunne logge ind igen uden først at have logget ud.

Proces

Planlægning af arbejdet

Til at starte med kiggede vi på de user stories der var blevet givet. Vi brugte dem til at lave os en oversigt over hvordan vi skal lave projektet. Vi indsatte user stories i et kanban-board, og lavede ud fra de mindre 'task', så vi struktureret kunne lave lidt af gangen, for at få en god start med at arbejde i teams på et projekt.

Vi lavede et domænemodel ud fra hvad der så ud til at være brug for gennem user stories, så vi vidste, hvilke Java klasser der skulle være. Der blev så lavet et ER-diagram, så vi hensigtsmæssigt kunne lave en database. Efter den var lavet, begyndte vi at lave klasser i vores IntelliJ-projekt.

Der var enighed om at splitte gruppens medlemmer op i back- og frontend grupper. Derfor begyndte nogle at lave frontend design i HTML og CSS.

Udførsel af arbejdet

De andre medlemmer begyndte at sætte thymeleaf op og lavede routing med Javalin i vores Controller-klasser, så vi kunne teste login-funktionen og gøre vores forhåndsvisning af vores design til en funktionel hjemmeside.

Nogle af funktionerne som skal tillade en bruger at ændre på databasen var en udfordring, så de har udgjort hoveddelen af arbejdstiden for de gruppemedlemmer som arbejder med backend-delen.

Løbene fik vi ikke opdateret og brugt vores kanban-board, da arbejdet flød af sig selv for både front- og backend-grupperne. Frontend-delen viste sig, som vi havde forestillet os, at

være lettere end backend-delen, så en af frontend-udviklerne begyndte efter et par dage at deltage i backend-udviklingen.

Med en relativ god mængde tid igen var back- og frontend ved at være færdigt, så vi startede roligt med at skrive rapporten samt at finpudse vores kode.

Hvad vi har lært

Hele forløbet er gået godt i forhold til hvor meget tid vi skulle bruge. Vi endte ikke med at være under tidspres.

Kanban-board
Måden vi arbejdede på fulgte dog ikke fuldstændigt vores forestilling om hvordan det ville være fra starten af projektet. Kanban-boardet viste sig at være unødvendigt og muligvis lidt bøvlet, da vi konstant ville skulle opdatere det og afbryde vores personlige workflow. Vi foretrak alle at arbejde ud i en køre med den overordnede user story vi hver især havde taget. Måske kan kanban-board virke godt i et andet projekt, eller for en anden gruppe.

Merge conflicts og fordeling af arbejde
Vi lagde løbene mærke til at merges af branches nogle gange gjorde at man i oversigten over merge conflicts skulle vælge at beholde én version, af to forskellige stykker kode, som to udviklere havde lavet samme sted. Det skete fordi frontend og backend udviklere begge havde brug for at lave HTML nogle gange. Backend udvikleren var inde i HTML klasserne for at lave thymeleaf funktionalitet.

Det var lidt bøvlet at skrive det hele sammen i ‘result’ afsnittet i merge conflict vinduet. Vi valgte at beholde thymeleaf funktionaliteten i stedet for den styling, der var lavet. Stylingen kunne vi hurtigt lave igen ved at kopiere og ændre små ting fra andre lignende HTML klasser, men det er ikke sikkert at det vil gå så nemt en anden gang.

Med andre ord, så endte man ved uheld med at overskrive noget af hinandens kode. Det var heldigvis et mindre problem. Vi udledte dog ud fra det, at det i fremtiden kunne være mere effektivt at backend udviklerne kun sætter knapper, forms, labels og andet op, og derefter hjælper backend udviklere med løsning af de metoder som giver problemer tidligt i projektet. Derefter vil de kunne lave stylingen.

Demo af program

Link til vores lille demo af vores program