Cupcake Projekt

Klasse B

- August Christoffer Bredahl Duelund - Github: AugustDuelund - Cph-mail: cph-ad228@cphbusiness.dk
- Mathias Filtenborg Hansen - Github: MathiasFHansen - Cph-mail: cph-mh914@cphbusiness.dk
- Mathias Egebjerg Jensen - Github: MathiasJensen96 - Cph-mail: cph-mj839@cphbusiness.dk
- Thias Meyer Petersen - Github: thiasmp - Cph-mail: cph-tp208@cphbusiness.dk

Indledning

Dette projekt omhandler at kreere en hjemmeside der har til formål at man kan gå ind som kunde og bestille cupcakes ud fra valgmuligheder for topping og bund. Som kunde kommer man ind på forsiden og kan så oprette sig via. en email adresse, som i dette tilfælde ikke behøver være gyldig, samt et password. Efterfølgende kan man så tilføje cupcakes til sin indkøbskurv og gå til betaling. Som admin kan man logge ind på en hardcoded 'employee' bruger hvorpå man kan indsætte penge på en kundes konto så kunden kan betale for sine cupcakes.

Baggrund

Dette program er lavet til Olsker cupcakes der er placeret på Bornholm. De lever af at sælge cupcakes til folk. De vil gerne udvide deres forretning og det har de valgt at gøre igennem et website. Det website har de nogle krav til.
De vil have at websitet skal:

- Kunde skal kunne oprette en konto
- Kunde skal kunne gemme ordre
- Kunde skal kunne bestille en eller flere cupcakes bestående af en bund og en top
- Kunde skal kunne fjerne cupcakes fra indkøbskurv hvis de fortryder eller har valgt forkert.
- Kunde skal kunne se samlet pris på cupcakes.
- Administrator skal kunne indsætte beløb til kundekonto.
- Administrator skal kunne se tidligere ordrer

Teknologivalg

- MySQL database - version 8.0.22
- Tomcat - version 9.0.44
- IntelliJ Ultimate edition - 2021.1
- JDBC - version 5.1.49
- SDK - Java version 1.8.0_261
- Bootstrap
- CSS
- HTML
- javascript
- Adobe XD

Krav

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 en ordre.

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

US-4: Som kunde kan jeg se mine valgte ordrelinjer i en indkøbskurv, så jeg kan se den samlede pris.

US-5: Som kunde eller administrator kan jeg logge på systemet med e mail og kodeord. Når jeg er logget på, skal jeg kunne 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 ordre fra min indkøbskurv, så jeg kan justere min ordre.

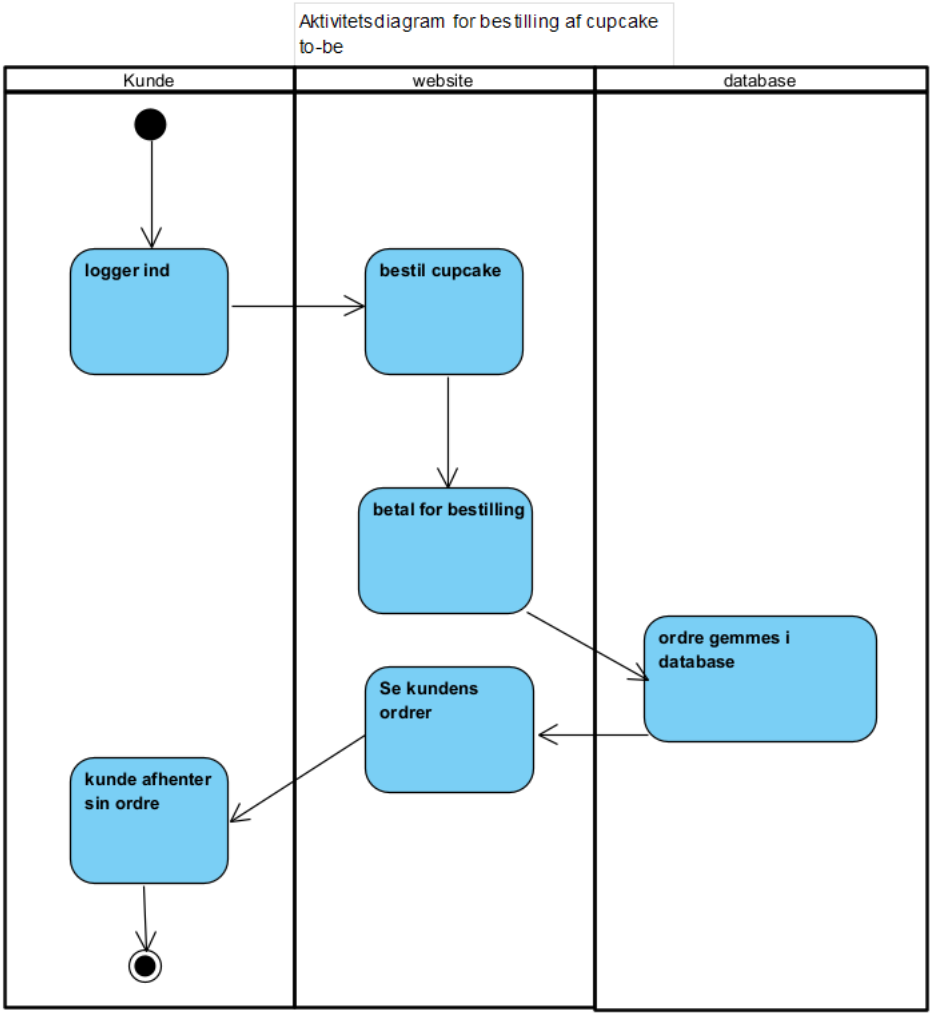
US-9: Som administrator kan jeg fjerne en ordre, så systemet ikke kommer til at indeholde ugyldige ordrer. F.eks. hvis kunden aldrig har betalt.

Aktivitetsdiagram

Vi har valgt at lave 2 aktivitetsdiagrammer da programmet er todelt, enten en employee del, eller en kunde del.

Kunde-diagram

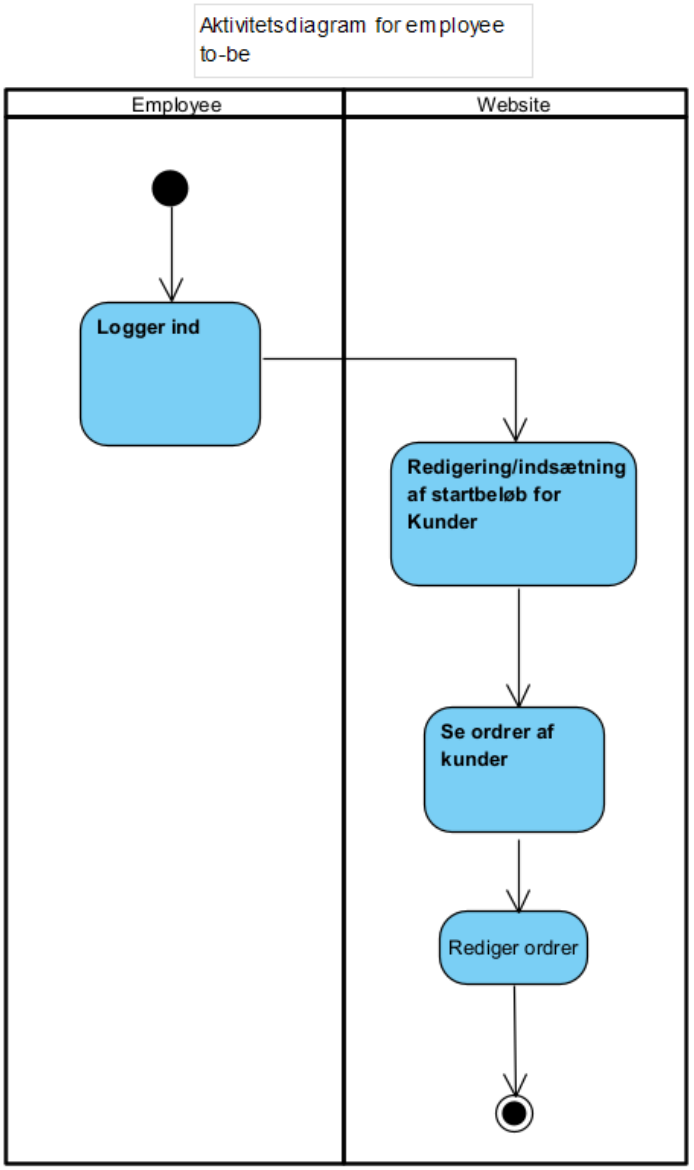
Vores første diagram viser hvordan en kunde bestiller en cupcake.
Først logger kunden ind og bliver ført videre til bestil cupcake, hvor kunden vælger den eller de cupcakes der ønskes. Derefter skal der betales for varen, hvor det bliver gemt i en database, hvor kunden kan tilgå sin/sine ordre. kunden skal derefter selv afhente sin ordre fysisk i Olsker cupcakes fysiske butik.



Employee-diagram

Vores andet diagram viser hvordan en employee kan holde styr på ordre gennem hjemmesiden. En employee kan logge ind på siden gennem systemet og kan derefter finde kunder ud fra deres 'id'. Employees kan så indsætte et beløb på kundens balance så kunden kan bestille cupcakes når de logger ind.

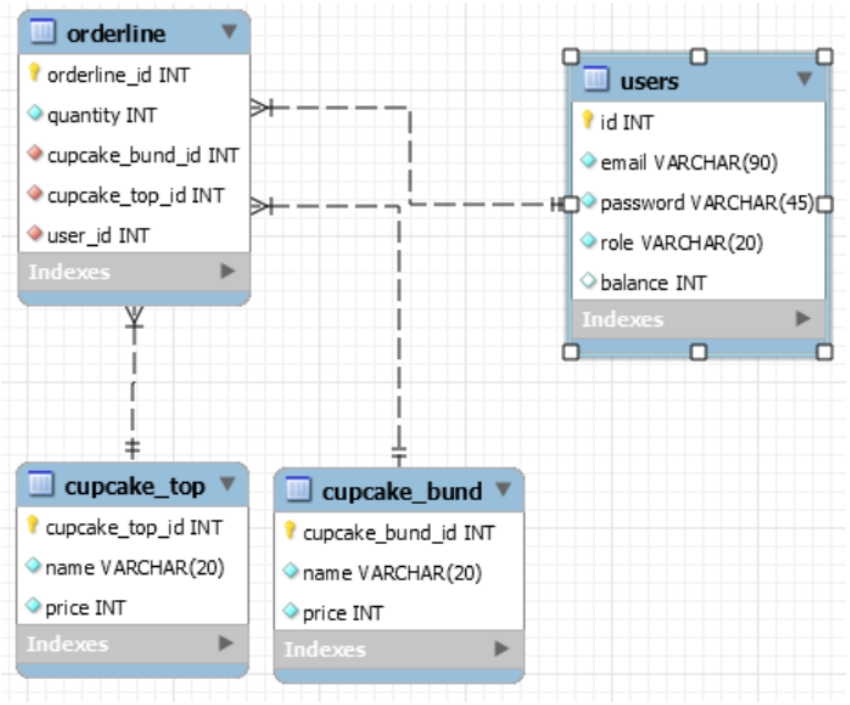
Derudover kan en employee så gå ind og finde en fuld liste af ordre og hvilke kunder der er tilknyttet dem eller finde en ordre for en enkelt kunde gennem kundens 'id'. Employees kan så hvis nødvendigt slette fra kundens ordre.



Domæne model og ER diagram

EER

For at danne os overblik over hvordan vi skulle igangsætte kodningen lavede vi et EER diagram til SQLdatabasen. I EER diagrammet har vi lavet 3 parent-tables: users, cupcake_top og cupcake_bund. I vores users table opbevarer en brugers email, password, rolle og saldo. Den data gemmer vi med i et 'id' som er primary key.
Dernæst har vi så både et table til cupcake top og bund, i disse tables er der hardcoded cupcake-navne samt priser som er gemt med unikke 'id' som er primary key.
Det hele bliver så samlet i et orderline table som modtager de 3 primary key 'id'er som foreign keys og laver et unikt 'id' til hver ordre når en kunde betaler for sin bestilling.

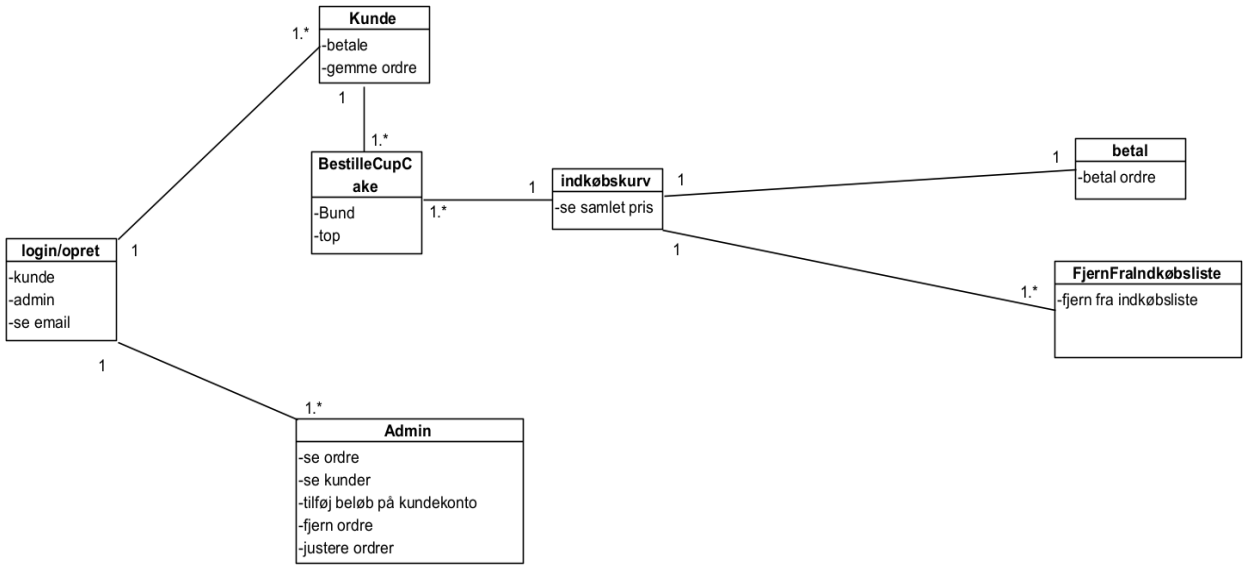


Domæne model

Vores domæne model viser en simpel forklaring på hvordan programmet er bygget op.
Vi starter med at kunne oprette en kunde eller admin, som derefter deler sig i forhold til om det er admin eller kunde der bliver oprettet.

Kunde kan betale for sine bestillinger, eller gemme sin ordre. kunden starter med at vælge at bestille en eller flere cupcakes, som er bestående af en top og en bund. Derefter vises en samlet pris for hans bestilling, hvor han enten kan betale ordren, eller fjerne en eller flere cupcakes fra hans bestilling.

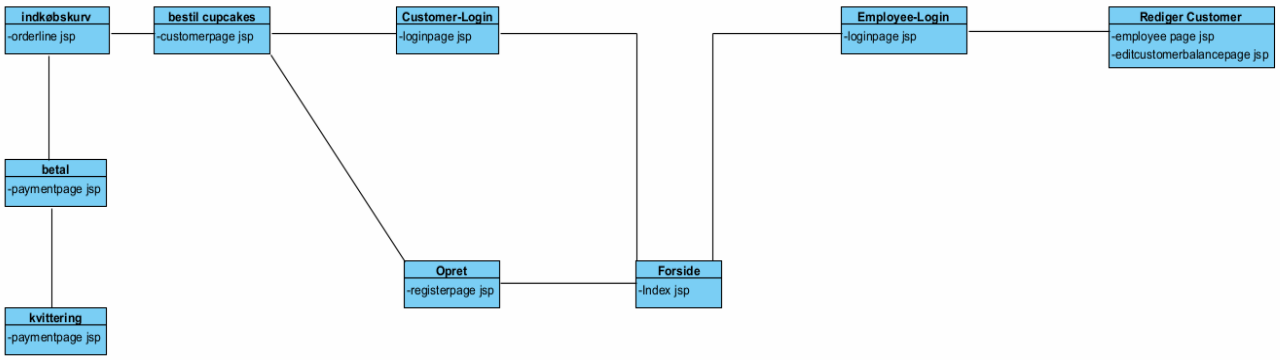
Admin kan se ordrer fra kunder, se kunder, tilføje beløb til kunder, fjerne ordre hvis de ikke er betalt for, så de ikke fylder, og kunne justere på ordrer.



Navigationsdiagram

Vi har benyttet os af en fælles navigationsbar på alle vores sider, der indeholder login knap, sign in knap, hjem til forsiden samt se din e-mail du er logget ind med.

Som tidligere har vi et splittet diagram, et der fører til kunde/customer siderne, og et der fører til admin/employee. Under alle de forskellige sider har vi skrevet de forskellige .jsp sider de forskellige dele af programmet bruger. Grunden til at opret/registerpage.jsp kun fører til bestil cupcakes er at vi ikke tillader at alle kan oprette en admin konti, men det skal gøres igennem mySQL.



Særlige forhold

- Hvilke informationer gemmes i session
  I vores system gemmer vi users på sessionscope når de logger ind så de kan blive benyttet indtil session bliver lukket.
  Derudover gemmer vi vores indkøbskurv på session så den ikke forsvinder hvis en kunde går over på f.eks. forsiden og vender tilbage til kurven.

- Hvordan håndterer man exceptions. Det kommer vi tilbage til senere i semesteret.
  Vi har benyttet exceptions til at håndtere når en employee vil indsætte et beløb på en kundes konto. Der skal ikke kunne indsættes et negativt beløb og det beløb han indsætter lægges oveni det beløb der i forvejen ligger i stedet for at overskrive det. Til det har vi bare brugt userexception.
  Ellers har vi brugt exceptions til at sørge for at en user ikke kan betale for en ordre de ikke har råd til.

- Hvordan man har valgt at lave sikkerhed i forbindelse med login.
  Der er ikke som sådan noget sikkerhed udover at brugeren skal eksistere i databasen. Hvis en tilfældig person skrev email og password til employee ville han kunne logge på som employee.

- Hvilke brugertyper, der er valgt i databasen, og hvordan de er brugt i jdbc
  I vores system har vi en customer/bruger rolle og en employee rolle. I systemet bliver de brugt til sortere hvilken side der bliver vist. Hvis man logger ind som kunde bliver man henvist til en kundeside, hvorimod hvis man logger ind som employee bliver man henvist til en employee side. Derudover har vi brugt kunderollen til kun at indhente kunder fra databasen når en employee skal kunne indsætte et beløb på konto.

Status på implementation

Efter en uge med gruppearbejde er hjemmesiden fungerende men der er dele af userstories som mangler. Man kan både bestille og betale for sine cupcakes, dog kan ordren ikke gemmes - sidste del af userstorie 2.
Alt fungere some det skal i US3-5 dvs. at administrator kan indsætte beløb på kunders konti, kunder kan bestille og have overblik over ordre/pris og login for både kunder og employee er understøttet.
Det er på nuværende tidspunkt ikke muligt for administrator at se alle ordrer i systemet, det samme gælder kundens egne ordrer - der kan derfor ikke redigeres i dem.
Der var planer som involverede styling, men da vi prioriterede funktionaliteten nåede vi ikke i mål med det.

Proces

Efter vi fik udleveret opgaven snakkede vi kort i gruppen om hvordan vi ville gribe den an, og blev hurtigt enige om en retning vi ville gå. Vi brugte de to første arbejdsdage på at lave EER og adobe xd og gik derefter i gang med at sætte startkoden op..
Da vi kom i gang med kode delen på dag 2 gik vi i gang med at lave udarbejde userstories i rækkefølge men sad en del fast i kommunikationen mellem scopes, jsp og java. Så den første userstory, at kunne bestille og betale cupcakes, blev udskudt lidt mens vi begyndte at udarbejde userstory 3.
Vi fik ikke nogle problemer da vi implementerede userstory 3 men faldt så tilbage til userstory 1. Der havde vi stadig nogle problemer da nogle af de andre userstories var afhængige af at userstory 1 fungerede. Vi fik noget hjælp fra en underviser som fik os på rette vej og fik så implementeret userstory 1 og 4 i et hug. Vi kunne altså nu både putte cupcakes i en indkøbskurv, bestille dem og betale dem så det trak den rette mængde penge fra brugerens konto. Vi løb dog ind i problemer med at få gemt ordren i selve databasen hvilket vi endte med at bruge lang tid på uden at få gennemført.

I vores projekt virkede javakode for det meste ret velfungerende men vi rendte ind i nogle problemer når vi skulle have alle sprog til at kommunikere. Næste gang vil vi nok spørge endnu mere om hjælp så vi potentielt ikke sidder fast på én ting ligeså længe ad gangen.