2018

Cupcake Projekt Rapport

Gruppe: CJLM

MICHAEL D. PEDERSEN, CPH-MP447@CPHBUSINESS.DK, MICH561D
LASSE A. NIELSEN, CPH-LN239@CPHBUSINESS.DK, LASSE A. NIELSEN
J. CHRISTIAN RYGE, CPH-JR221@CPHBUSINESS.DK, RANGERRYGE

Indholdsfortegnelse

Indledning

I Cupcake Projektet skal der skal laves en hjemmeside, hvorpå der kan bestilles cupcakes med en topping og en bottom. Der vil ikke være mulighed for at alle kan komme ind og bestille cupcakes, hvorfor der laves en side som kræver login og password før en bestilling kan gennemføres. Brugerne skal derfor valideres ved at tjekke databasen, og findes kunden ikke, skal vedkommende oprettes og efterfølgende gemmes i databasen.

Er kunden oprettet bliver de videresendt til bestillingssiden, hvor der vil være to dropdown-menuer. I disse vælges en topping og en bottom, og prisen vises samtidig.

Kunden vil have mulighed for at vælge antallet af cupcakes og se den samlede pris. Udover dette, vil der være mulighed for at kunne bestille flere, hvilket kræver en cart hvori den forrige bestilling gemmes.

I det følgende er *kunden* virksomheden Cupcake Shoppen, og *brugeren* er denne virksomheds kunder.

Baggrund

Cupcake Shoppen har siden foråret 2018 bagt unikke cupcakes. Shoppen lægger særligt vægt på gode råvarer og økologi, og samtidigt bestræber de sig på at holde prisen nede til fordel for forbrugeren. Cupcake Shoppens mål er et produkt af høj kvalitet og en høj efterspørgsel. På nuværende tidspunkt har virksomheden haft en høj vækst og er derfor for store til kælderbutikken på Vesterbro i København. Som følge af dette ønsker virksomheden at strømline deres bestillingsproces og ekspandere ud af København, og derfor er dette projekt påbegyndt.

Kunden ønsker et stilrent og indbydende design, der byder besøgende velkommen. Ydermere er det vigtigt, at man får en fornemmelse af kagerne – *man skal nærmest kunne dufte kagerne fra det øjeblik, at man klikker ind på siden*. Der skal især fokuseres på brugervenlighed, og antallet af sider skal være få, så kunden ikke skal navigere for meget rundt på siden. Til slut ønsker virksomheden mulighed for selv, at kunne administrere og opdatere lager samt kundekartotek, hvilket skal kunne udføres uden programmeringserfaring.

Valg af teknologi

Vi har valgt at skrive programmet i Java. Derfor faldt valget på Netbeans, idet programmet effektivt håndterer de valgte programmeringssprog og understøtter Apache Tomcat, hvilket er et godt serverværktøj. Til opbygningen af databasen har vi valgt MySQL, da deres Workbench er et godt udviklet SQL-program.

Programmer:

- Netbeans 8.2
- MySQL Workbench 8.0
- Java & Java SE (JDBC) 181
- Apache Tomcat 8.0.27.0

Sprog:

- Java
- HTML
- CSS
- MySQL 14.14
- JavaScript

Domæne model og ER diagram

Domæne model

Bilag 1

Efter første møde med kunden, blev vi enige om følgende Domæne-model som efterkommer kundens ønsker. Kunden lagde især vægt på udviklingen af et brugerkartotek, hvori der kunne lægges navn, loginoplysninger, mulighed for en online Wallet samt kontaktoplysninger – i dette tilfælde var eneste krav en e-mailadresse.

For at skabe overblik over produkter og tilvalgsmuligheder, forklarede vi vigtigheden af at af samle hver enkelt cupcake som ét produkt, idet hver kage er samlet af en top og en bund som hver har op til flere variationer.

For at kunne samle en kage skal der være mulighed for at vælge imellem forskellige Toppings og Bottoms, alt efter brugerens ønsker. Disse har forskellige priser, og dermed er det lettere at splitte dem op for at få en dedikeret ”Toppings-vælger”, og det samme gælder for Bottom. På denne måde kan brugeren vælge fra forskellige bunde og toppe og efterfølgende samle dem i produktet, hvor prisen for hver enkelt kage beregnes. Dette gør også opdatering og vedligehold væsentligt lettere, da man let kan tilføje elementer i hver kategori uden at samle alle variationer på forhånd.

Når brugeren har placeret en bestilling, er det vigtigt for kunden at få en oversigt over bestilte cupcakes for hurtig ekspedering samt sikre, at den korrekte bruger afhenter bestillingen. Ordredata for hver enkelt ordre skal gemmes, for at kunden nemt kan holde regnskab og oversigt over hvilke ingredienser der skal bestilles.

Før brugeren kan placere en ordre, vil hvert udvalgt produkt blive tilføjet i en indkøbskurv, hvor der er mulighed for enten at tilføje flere produkter, slette produkter eller tømme kurven helt. Dette gør brugeroplevelsen lettere og kan i bedste fald tilføje kunden øget omsætning, når det bliver lettere for kunden at tilføje andre variationer af produkterne. Hver indkøbskurv bliver knyttet direkte til en specifik bruger.

Ydermere er det et stort ønske for kunden, at de selv kan tilføje flere produkter og oprette nye medarbejdere med medarbejderrettigheder, og dette uden decideret programmeringserfaring. Derfor giver vi kunden mulighed for at udføre ovenstående igennem administratorrettigheder på siden.

Klasse diagram

Se bilag 2

Ud fra Domæne-modellen, udarbejdet med kunden, har vi udtænkt følgende diagram. Vi begyndte med at oprette databasen for kunden, da produkter og brugeroplysninger nemt skal lagres, opdateres og slettes efter behov. For at kunne tilgå databasen fra Java, har vi lavet en databaseforbindelse og dataforbindelses-objekter som kan hente og skrive til databasen. Næste trin i processen er at få lavet en korrekt tre-lags arkitektonisk vej fra databasen til det logiske lag og videre til præsentationslaget. Derfor er en kontroller som arbejder med data samt databasen valgt. For at kunne videregive data til websiderne, skal vi have en ny kontroller, som har adgangen til vores præsentationslag. Denne kontrollers opgave er at tage imod input fra brugeren, samt kunne vise valgt data for brugeren. Siden der skal tages højde for, at kundens brugere er mennesker, og dermed kan lave input fejl, skal det sikres, at der bliver valideret på brugerens input, når de registrerer sig i systemet, og når de logger ind.

Når brugeren er implementeret i systemet, arbejdes der videre med at udarbejde selve butiksdelen. Denne del er også opbygget af en tre-lags arkitektonisk metode, hvori der i præsentationslaget er forskellige sider brugeren kan interagere med systemet. Altså kan brugeren tilgå data som systemet henter fra databasen.

Kundens ønske om at kunne administrere systemet selv uden programmeringserfaring medførte implementeringen af administratorrettigheder og webside. Denne del bliver tildelt en allerede eksisterende bruger som udvider brugerprofilen.

For at få bundet alle dele sammen og samtidig give brugeren et overblik over websiderne, er der lavet en index/startside hvor brugeren nemt kan navigere fra.

Der er også planlagt følgende elementer som let vil kunne implementeres i systemet ved behov. Først er der planlagt en hurtigere og bedre metode til at hente produkter fra databasen og præsentere dem på websiderne. Herefter fulgte at gemme de valgte produkter fra brugerens indkøbskurv i databasen, så denne gemmes selvom sessionen afbrydes. For at udvikle videre på denne del af systemet, skal der også kunne gemmes ordre i databasen. Slutteligt vil der implementeres en betalingsside.

ER diagram

Se bilag 3

Først vil vi forklare det mulige brud på første normalform, da vi har valgt ikke at dele navnet op i tre dele, men derimod har beholdt det som én. Dette vil for nogle betyde, at der er multivalues i databasen. Vi mener at have opfyldt kundens behov, idet vedkommende ikke har behov for at sortere på alle tre dele, men blot fornavn for afhentning af ordre.

Databasens tabeller bruger ikke automatisk genereret ID. Grunden til dette er, at vi ikke har haft tid til både at kunne ændre det i databasen og i vores kode. Vi er bevidste om fordelene og planlægger at implementere dette snarest. Vi har overvejet at optimere vores ER-Diagram, da både Cart og Order indeholder samme data. Derfor vil det kunne betale sig at flage den, når den er købt. Det er tydeligt at vores ER-Diagram ikke er tilpasset vores Domæne-model, da vi mangler visse data som burde lagres i databasen. Disse ændringer bliver lavet snarest.

Navigationsdiagram

Se bilag 4

Når en bruger navigerer til websiden, ankommer de til startsiden hvorfra de kan navigere rundt på resten af siden. Men da kunden ønsker at brugere skal logge ind, er det først muligt at komme videre til butiksdelen efter login. Hvis man endnu ikke har en bruger, bliver man tilbudt at oprette sig for derefter at kunne navigere videre på siden. Vi tager forbehold for tastefejl og ugyldige kriterier. Dette tydeliggør diagrammet ved blandt andet at registrere Refererer til sig selv, hvis brugeren indtaster ugyldige data. Ligeledes vil loginsiden ikke modtage ugyldige data. Skulle det være tilfældet, vil det blive pointeret for brugeren, som vil blive tilbudt at oprette en bruger, hvis de ikke allerede har en. Når brugeren er logget ind, vil de blive sendt videre til butikken, hvor de har mulighed for enten at handle eller gå til deres personlige brugerprofil.

I toppen af hver side har vi lagt en navigationsbar hvorfra en bruger kan navigere til alle åbne sider. De lukkede sider, såsom indkøbskurv samt betalingssiden, kan man kun komme til gennem en direkte reference fra shoppen. Efter kundens ønske er der også lavet en administratorside, hvilket også kan tilgås fra navigationsbaren. Denne kræver selvfølgelig administratorrettigheder. Hvis den pågældende bruger ikke har disse rettigheder vil de blive returneret til den foregående side. Dette simplificeres i diagrammet da den eneste synlige vej til administrator siden går gennem brugersiden.

Sekvens diagram

Se bilag 5

Dette diagram viser sekvensen når en kunde har valgt en cupcake og ønsker at gennemføre et køb. Efter brugeren klikker på ”Læg i kurv”-knappen, vil systemet indsamle de data som brugeren har efterspurgt og dermed vil vores Frontcontroller sende bud efter Controlleren, som først vil hente den topping brugeren vil have fra databasen. Dernæst vil Frontcontrolleren sende bud efter Controlleren igen, for at hente bundens data fra databasen. Når Controlleren har hentet både bund og topping samt sendt data tilbage til Frontcontrolleren, vil Frontcontrolleren sende den hentede data fra databasen videre til en ny side på systemet, hvor brugeren nu kan se den udvalgte vare.

Brugeren har nu to valgmuligheder: 1) Brugeren kan vælge at gå tilbage til shoppen hvorved sekvensen starter forfra ved at kunden vælger topping, bund og antal eller 2) brugeren kan vælge at købe indholdet fra sin indkøbskurv, hvorved brugeren, i stedet for at komme tilbage til shoppen, vil komme til betalingssiden hvor brugeren betaler og efterfølgende kan hente sin ordre.

Dette kunne gøres smartere og mere effektivt ved at lade Controlleren gå i databasen to gange men kun returnere én gang i stedet for de to gange den gør på nuværende tidspunkt. Dette løses ved at bruge den allerede eksisterende men ikke-implementerede productDTO.

Særlige forhold

Når en bruger logger ind, gemmes brugerens username i sessionen for at sikre at brugeren rent faktisk er logget ind når de besøger butikken. Enkelte steder i koden bruges exceptions til at kontrollere, om der forekommer input-fejl. Dette kan ses i metoden handleLogin som findes i Frontcontroller klassen. Måden vi bruger exceptions på er ved at forsøge at hente et parameter fra requestet. Hvis dette fejler findes pågældende parameter ikke, og dermed sendes brugeren til login. Fejler det derimod ikke har brugeren allerede forsøgt at logge ind, hvilket er mislykket. Herefter sendes brugeren til loginsiden med parameteret error, hvilket gør det muligt for systemet at fortælle brugeren at loginforsøget er ugyldigt.

Når der skal oprettes en bruger, tjekker systemet før oprettelsen af brugeren i databasen, at der er blevet indtastet gyldig data. Systemet validerer først brugernavnet ved at tjekke om det er udfyldt og sikrer sig at der ikke er semikolon i navnet, samtidig med at længden af brugernavnet er inden for kundens kriterier.

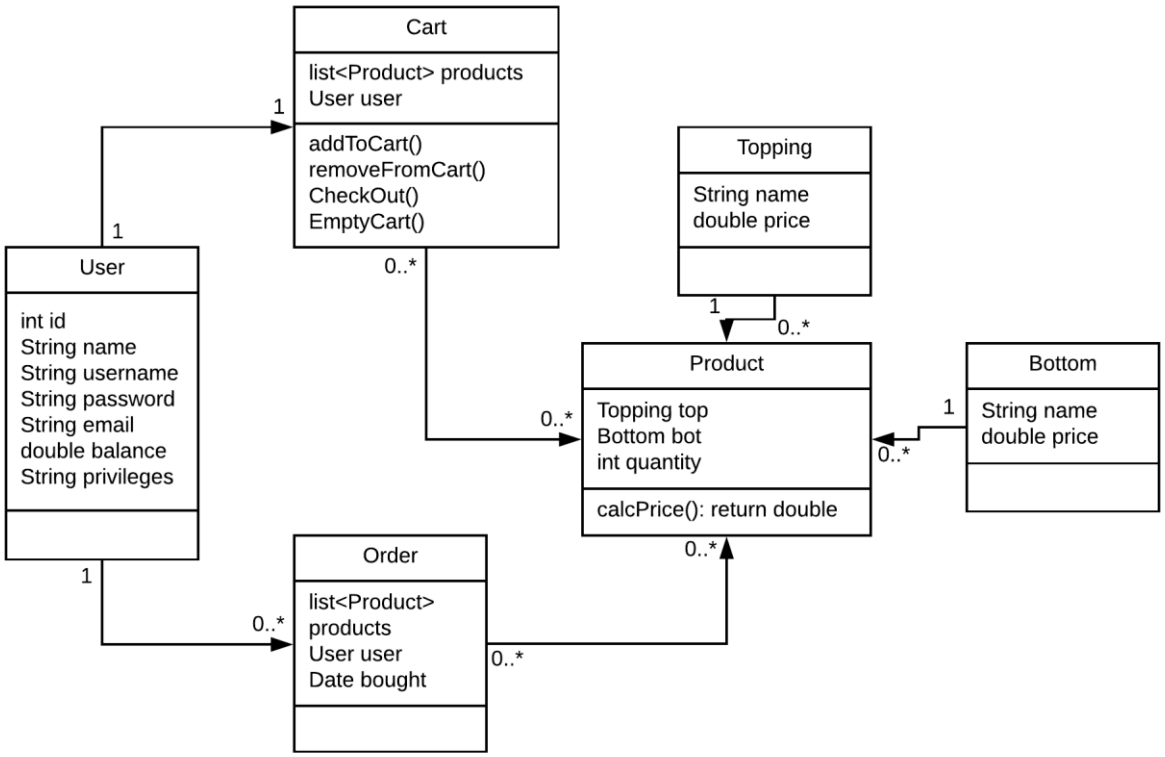
Bliver brugeren sendt videre, sikres det også at brugernavn ikke allerede eksisterer. Efterfølgende tjekkes passwordet, som skal gennem næsten samme type validering. Kunden mener, at det skal være tilladt for flere brugere at have samme password, og derfor tester systemet ikke om det allerede findes i databasen. Til sidst i oprettelsen tjekkes om der er blevet indtastet en korrekt formateret e-mail. Til dette bruges samme fremgangsmåde som før, men denne gang kalder systemet også en ekstern e-mailvalidator som tjekker formatet. Til sidst tjekkes også databasen om en allerede eksisterende bruger har samme e-mail, da kunden ikke vil knytte samme e-mail til flere brugere. Når en bruger skal logge ind, validerer systemet også inputtet. Systemet tager brugernavnet og sikrer sig på stort set samme måde som i registreringen, men her vil det også tjekke om brugeren med det pågældende brugernavn har samme password som det brugeren har skrevet. Det gøres ved at hente den gemte bruger op fra databasen og sammenligne de to passwords, og er det det korrekte password der er skrevet, får man lov til at blive logget ind. Under validering af registreringen samt ved login tjekkes det, om der er semikolon med i inputet, for at sikre at folk ikke prøver at lave et metodekald. På nogle af kaldene til databasen er der lavet "prepared statements" som sikrer, at brugerens input ikke laver problemer for systemet.

Status på implantation

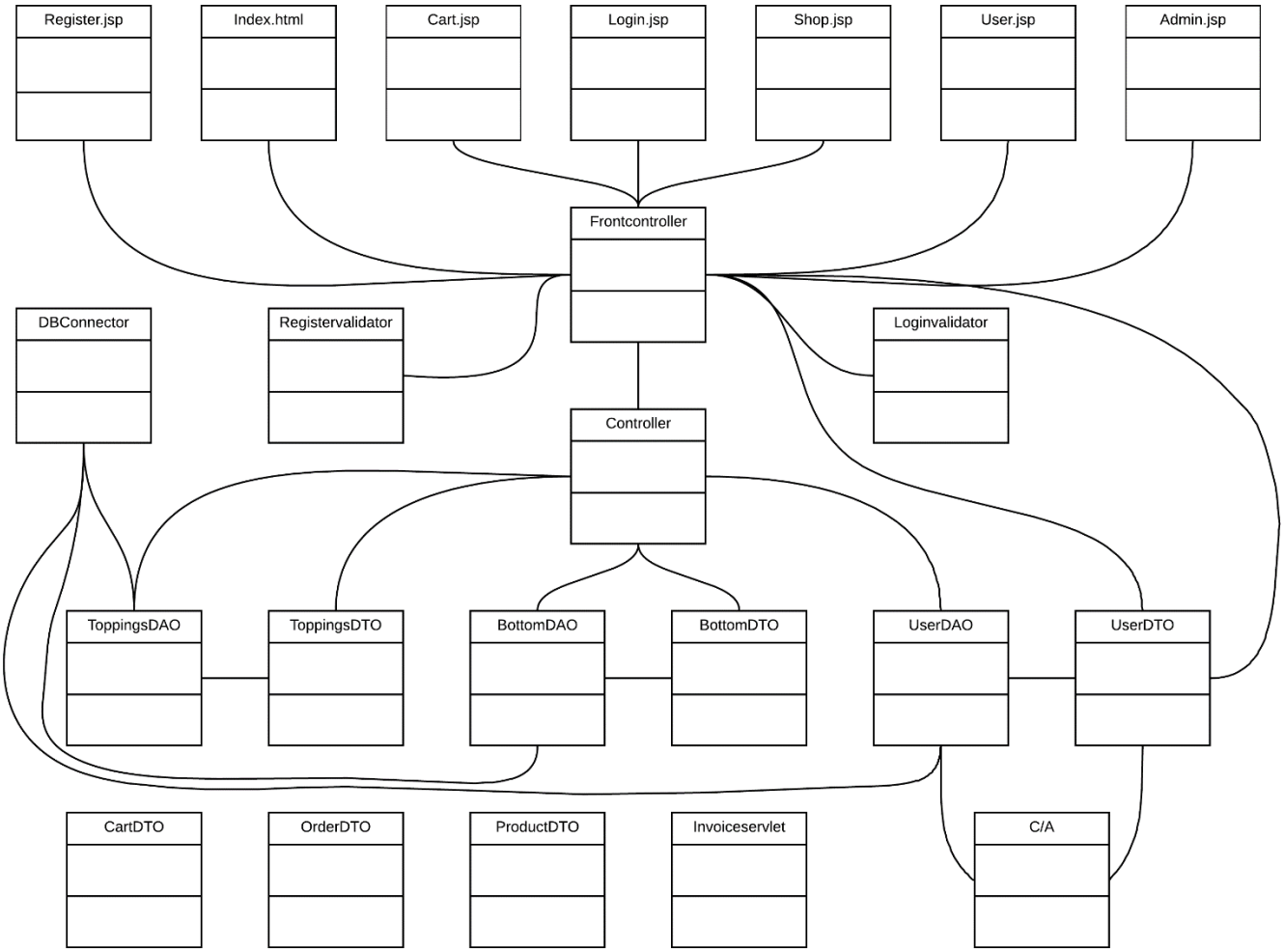
Der er stadig visse mangler på et par af websiderne. På brugerens profilside mangler der noget funktionalitet, hvor man kan ændre sine personoplysninger samt ændre adgangskode til brugeren. Man skal også kunne opfylde sin Wallet. Desuden skal man på brugerens profilside også kunne se tidligere ordrer. Betalingssiden er endnu ikke færdiggjort, idet den skal vise et hurtigt overblik over ordren og indholdet af brugerens Wallet. Der skal være en knap som, når man klikker på denne, lader ordren gå igennem samtidig med den gemmes i databasen. Adminstratorsiden er endnu ikke blevet implementeret. På denne side skal administratorerne kunne slette/tilføje toppings og bunde, slette brugere eller redigere brugernes rettigheder. Da alle websiderne ikke er lavet, mangler der derfor også CRUD-metoder (Create, Retrieve, Update og Delete) til de tilsvarende websider. En kort opsamling af metoderne er: lav topping, lav bund, update brugerens oplysninger samt password og Wallet, samt ændre rettigheder for brugere og slet bruger, topping eller bund. Der skal også laves CRUD-metoder til indkøbskurven, både 'udføre'- og 'hente-metoder', hvilket også gælder for ordrer. Der er stylet lidt på alle siderne, men yderligere tid kunne godt være blevet brugt på dette.

Der opstår fejl når man prøver at komme ind på brugerprofilsiden. Fejlen ligger i en NumberFormatException. Desuden bliver man ved en fejl sendt til startsiden, når man prøver at navigere til indkøbskurven.

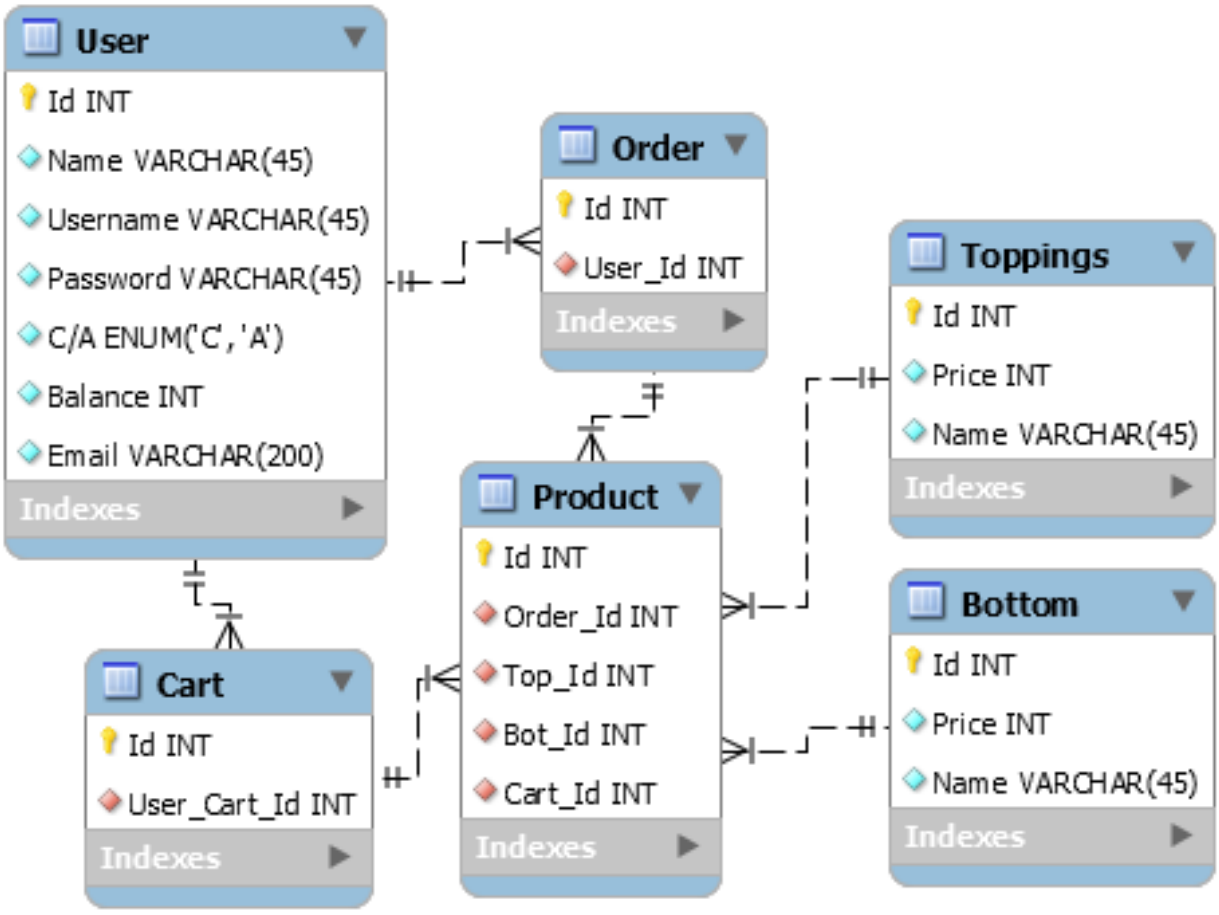
Bilag 1: Domæne-model



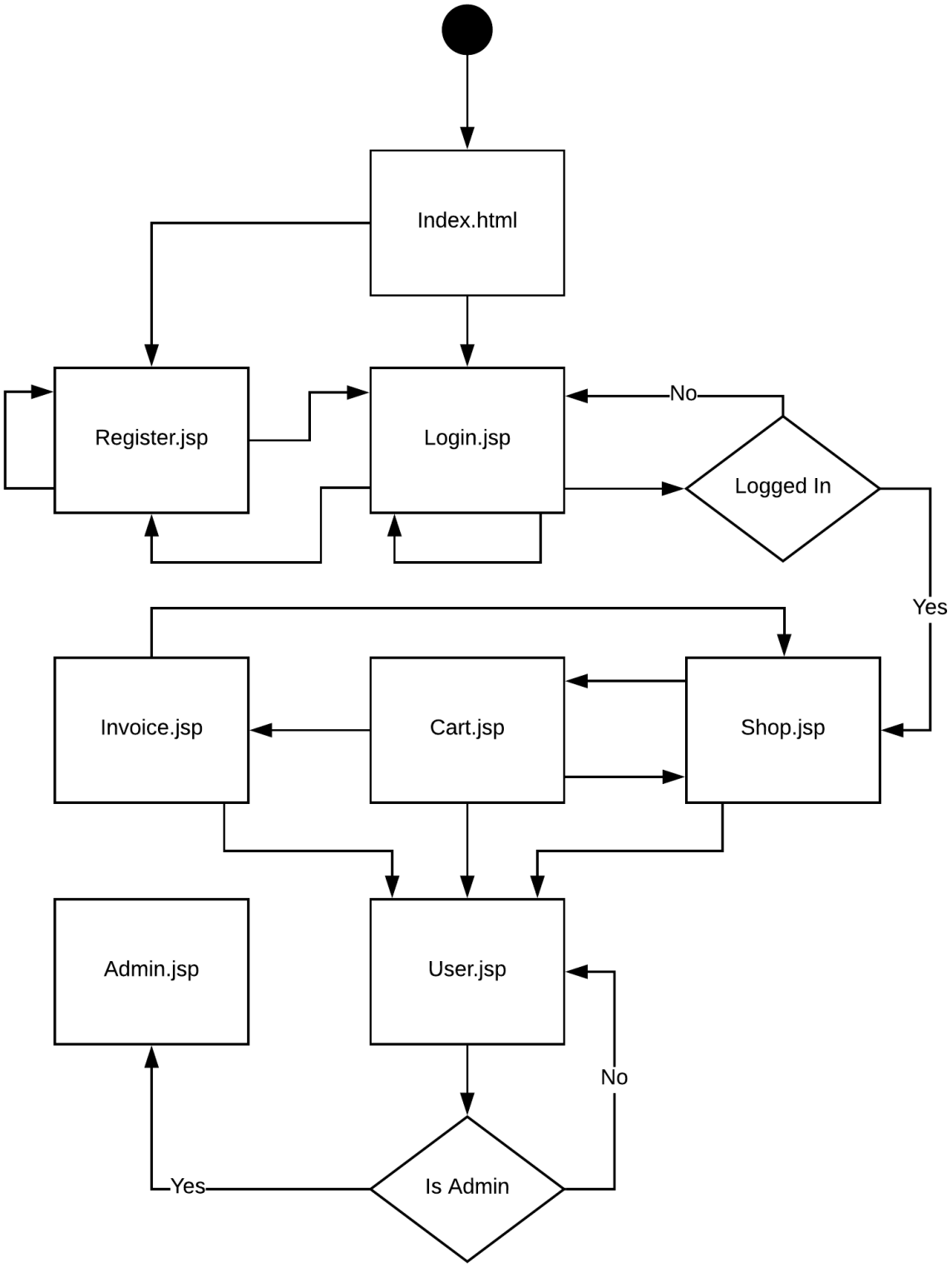
Bilag 2: Klasse Diagram



Bilag 3: ER Diagram



Bilag 4: Navigations Diagram



Bilag 5: Sekvens Diagram