Projekt Cupcake

23/10 → 02/11 - 2023

Lavet af

Ahmad	cph-aa540@cphbusiness.dk	@Babanoelk
Hanni	cph-ha186@cphbusiness.dk	@HanniSalman
Lasse	cph-lh225@cphbusiness.dk	@Lassekh
Youssef	cph-yb48@cphbusiness.dk	@Badranyoussef

Indholdsfortegnelsen

Indledning

Med stor interesse i Full-Stack / Web Udvikling springer vi ud i en udfordrende rejse for at dykke ned i hvordan man udvikler webshops.

Teamet bag dette projekt består af fire ambitiøse datamatikerstuderende på andet semester, som aldrig har kodet før studiestart. Kort efter studiestart havde vi alle en tanke om hvordan vi nogensinde vil kunne skabe noget i fremtiden. Sproget (Java) var i den grad ukendt for alle 4. Efter første semester faldt flere brikker på plads, og vi begyndte at se lyset for enden af tunnelen.

Denne rapport henvender sig til datamatiker studerende som er på eller har afsluttet anden semester.

I skrivende stund er vi på andet semester hvor vi har fået til opgave at bygge en CupCake webshop hvor back-end kodes i Java, frontend i HTML. For at skabe en sammenhængende applikation, har vi valgt at bruge Javalin som framework og Thymeleaf som skabelon motor til at koble backend og frontend sammen.

Formålet med denne rapport er at beskrive vores arbejde med at udvikle en CupCake webshop, hvor vi har fokus på kundeoplevelsen. Dette gør vi ved at give kunden muligheden for at udforske siden og tilføje cupcakes til sin kurv uden at være logget ind. Når kunden ønsker at foretage en bestilling, bliver kunden bedt om at logge ind med en eksisterende bruger eller oprette en ny bruger.

Baggrund

Olsker Cupcakes et dybde økologisk iværksættereventyr fra Bornholm, som har ramt den helt rigtige opskrift. Virksomheden ønsker en simpel webshop til salg af CupCakes.

Kunden ønsker at besøgende på webshoppen skal kunne bestille cupcakes ud fra sammensætningen af topping og bund som virksomheden. Besøgende skal kunne oprette en bruger, så de kan gennemføre en ordre og se en liste over tidligere ordrer. Kunden ønsker at kunne logge ind som administrator for at se et overblik over bestillinger og kunder.

Teknologi valg

Til denne opgave har vi brugt følgende teknologier:

- IntelliJ 2023.2.3
- Java v. 17
- Postgres
- Javalin
- Thymeleaf
- JDBC
- HTML
- CSS

Krav

Målet med Olsker Cupcakes' online bestillingssystem er at forbedre kundeoplevelsen, ved at tilbyde nem og bekvem bestilling af cupcakes. Dette system har til formål at:

1. Gør det lettere for kunderne at bestille cupcakes fra hjemmet eller enhver anden placering.
2. Øge bekvemmeligheden ved at undgå ventetid i bageriet.
3. Øge salget ved at nå ud til en bredere kundebase.
4. Hjælpe med lagerstyring og reducere spild.
5. Indsamle data om kundepræferencer og salgsmønstre for at tilpasse menuen.

User stories:

US-1: Som kunde kan jeg bestille og betale cupcakes med en valgfri bund og top, så 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 ordrelinjer 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).

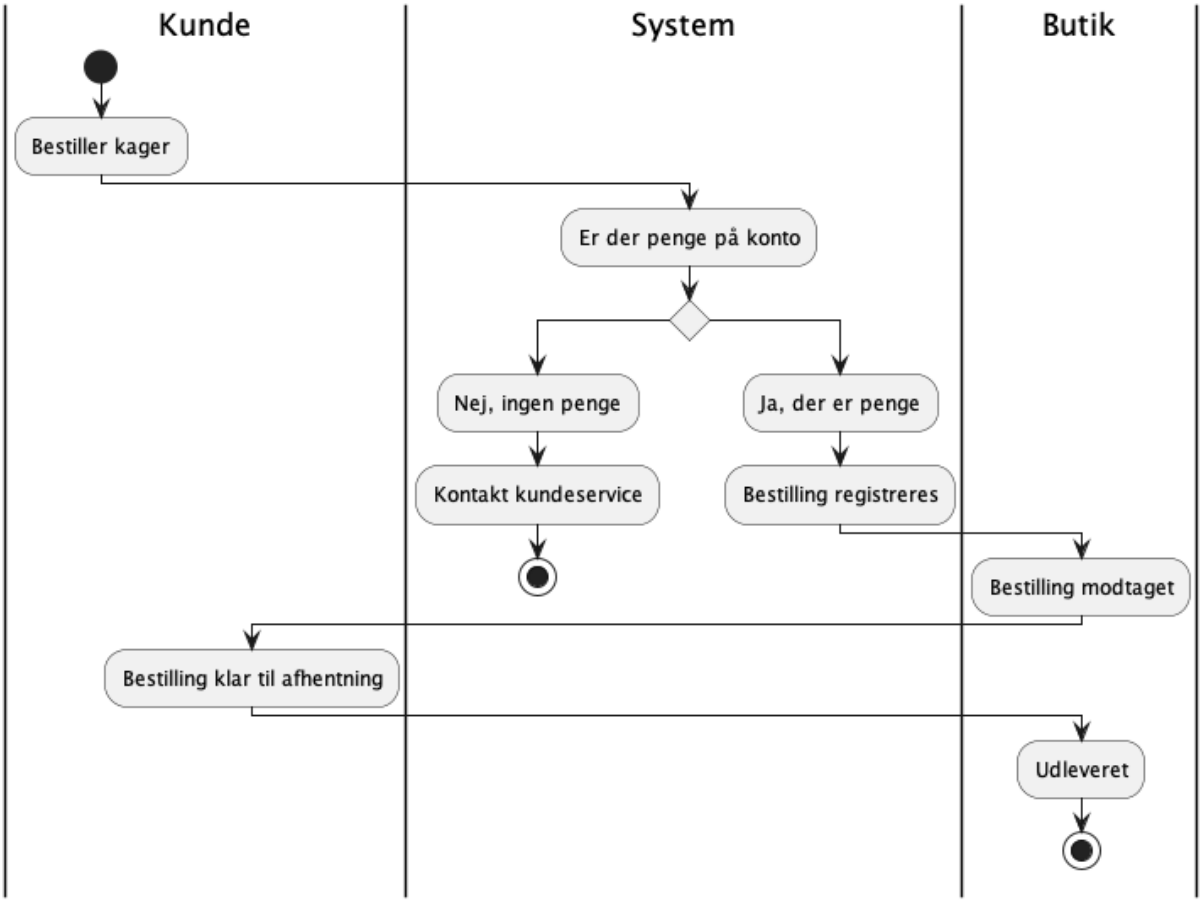
US-6: Som administrator kan jeg se alle ordrer i systemet, så jeg kan se hvad der er blevet bestilt.

US-7: Som administrator kan jeg se alle kunder i systemet og deres ordrer, så 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



[1]

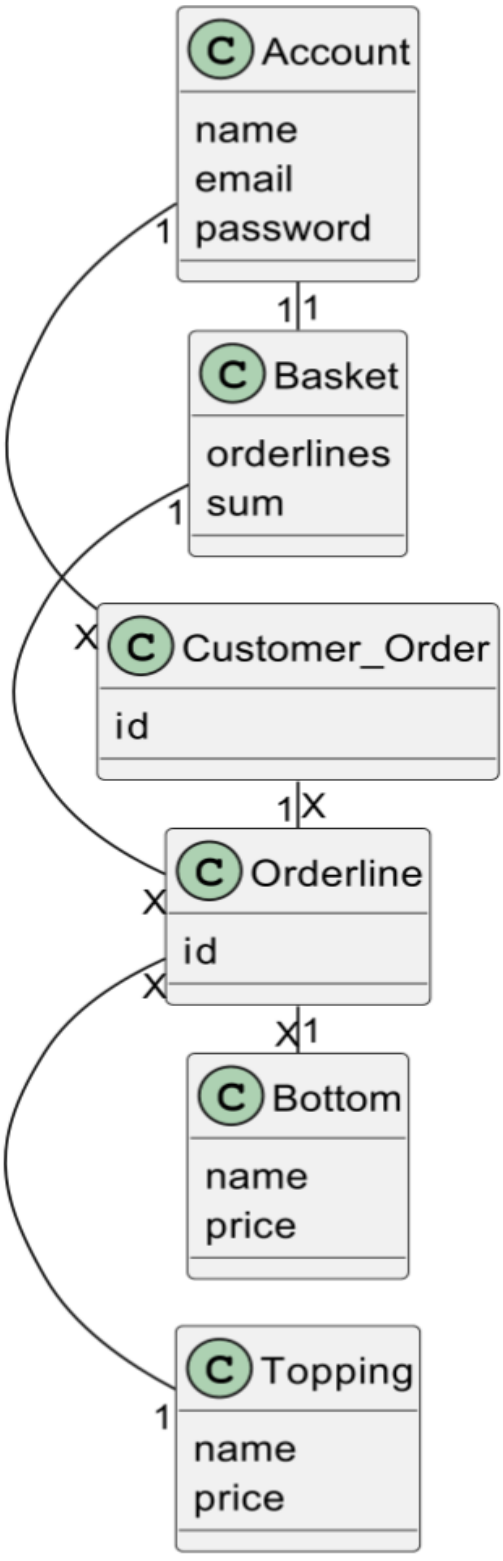
[1] Bilag 1

Domænemodel

Domænemodellen viser et system, der sammen skal håndtere et online-system til bestilling af kager. Modellen indeholder forskellige klasser, der ved hjælp af hinanden skal opfylde målet. [2]

Her bruges følgende notation: 1 … *

For øjnenes skyld har vi i stedet for en lille stjerne brugt et stort X.



Uddybning af hver klasse ses forneden:

Account

Denne klasse repræsenterer en bruger.

Klasse indeholder følgende informationer om brugeren:

- Navn
- E-mail
- Adgangskode

Hver bruger har kun en indkøbskurv og kan have mange kundeordrer.

Customer_Order

Denne klasse repræsenterer en kundeordre.

Normalt bruges et unikt identifikations-ID kun i EER-diagrammer, men i dette tilfælde har vi valgt at inkludere det, da det afspejler en almindelig praksis, som en bruger oplever i den virkelige verden.

Denne klasse indeholder følgende informationer om kundeordreren:

- ID

Hver kundeordre kan have mange ordrelinjer.

[2] Bilag 2

Orderline

Denne klasse repræsenterer en ordrelinje, som vi kender fra den virkelige verden. Nøjagtigt som på en kvittering.

Denne klasse indeholde følgende informationer om ordrelinjen:

- Top
- Bund

Hver ordrelinje har mindst en top og en bund.

Topping

Denne klasse repræsenterer toppen af kagen.

Denne klasse indeholder følgende informationer om toppen:

- Navnet på smagen
- Pris

Bottom

Denne klasse repræsenterer bunden af kagen.

Denne klasse indeholder følgende informationer om bunden:

- Navnet på smagen
- Pris

Basket

Denne klasse repræsenterer en kurv, som vi kender fra den virkelige verden, som vi bruger, når vi sætter ting i den.

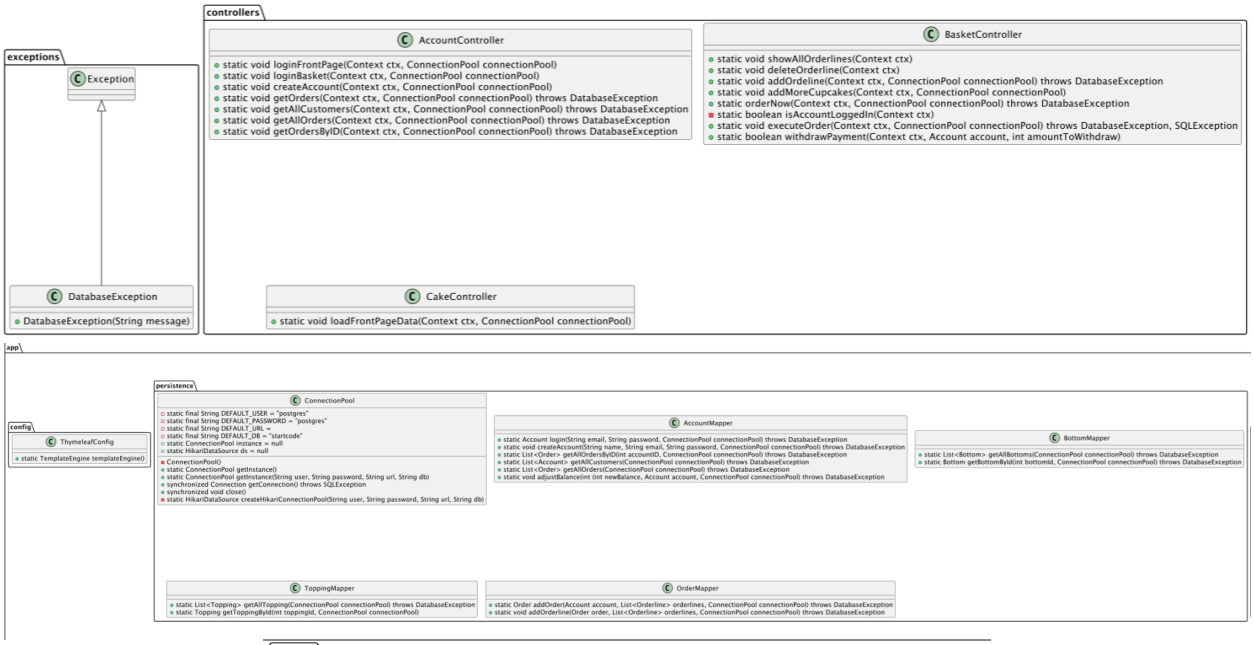
Denne klasse indeholde følgende informationer om kurven:

- Ordrerlinjer
- En samlet pris

Der findes kun en kurv til hver bruger, som de er tilknyttet til.

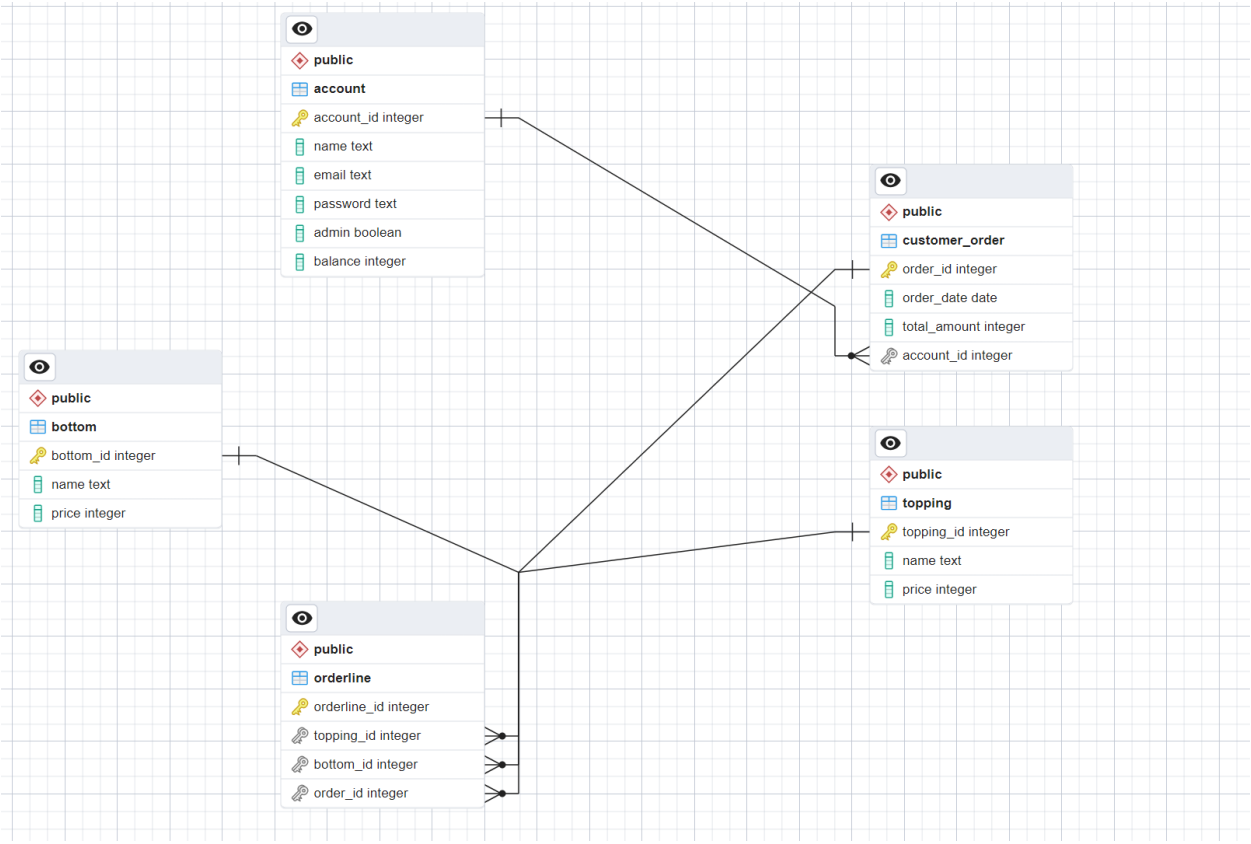
Klassediagrammet

Forneden ses vores færdigudviklede klassediagram. For et mere overskueligt billede af klassediagrammet, tjek linket på siden “Bilag”. [3]



[3] Bilag 3

EERD



4

[4] Bilag 4

Mappestruktur

Kodebasen er organiseret på en klar og struktureret måde for at sikre at udviklere nemt kan navigere i projektet. Vi har oprettet mappen “app” som indeholder backend-koden og "resources" mappen som indeholder frontend-delen af projektet.

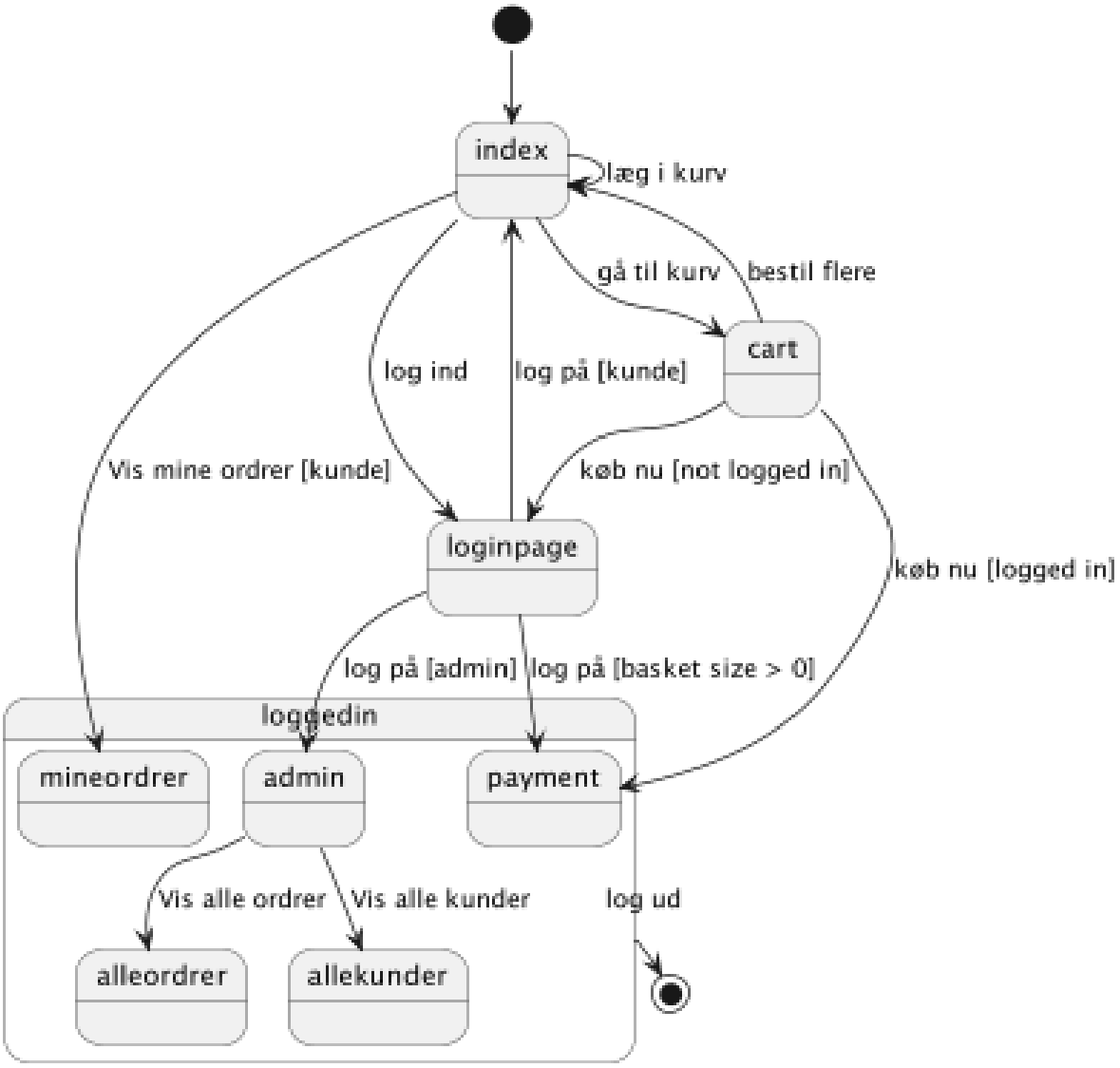
- Config - Her ligger konfigurationsfiler og indstillinger
- Controllers - Mappen indeholder kontrol logikken af objekter i systemet.
- DTOs - Burde indeholder dataoverførsels-objekter. Vi glemte at implementere DTO’er i projektet
- Entities - Indeholder dataobjekter der repræsenterer
- Exceptions - Indeholder brugerdefinerede fejlhåndteringer og undtagelser
- Persistence - Indeholder mappers til datatilgang og datalagring
- Services - Mappen burde indeholder ”services” til at styre flowet af data mellem controller og mapper

Derudover har vi mappen “resources” som indeholder to mapper til vores frontend.

- Public
    - CSS - Indeholder stylesheets
    - Images - Indeholder billeder
- Templates - Indeholder frontend templates

Navigation Diagram

Her ses vores navigationsdiagram, der viser hvordan man kan navigere rundt på siden som bruger. Der er opsat guard conditions hvor kun en bestemt rolle kan få adgang, samt lavet et separat state for brugere der er logget ind, som afsluttes når man logger ud.[5]



[5] Bilag 5

Særlige forhold

I det følgende afsnit er der beskrevet hvilke særlige forhold der gør sig gældende for den version af systemet vi har udviklet.

Det er essentielt for systemets funktionalitet, at vi ved hvem der er brugeren. Du kan være en besøgende på hjemmesiden, men du kan også være en bruger, der er logget ind. En bruger kan yderligere have forskellige roller.

For at holde styr på hvem der har puttet noget i kurven, har vi lavet en “session basket” der sættes når hjemmesiden loades første gang. Vi har også valgt at benytte os af en “session account”, til at holde styr på hvem der er logget ind, foretager køb osv. Den sættes når…

Der er to typer brugere: kunde el. admin. Da kravet var at vi kun skulle håndtere de to typer brugere, er det angivet med en boolean kolonne i databasen, der er true hvis brugeren er admin og false hvis brugeren er kunde.

I forbindelse med login checkes der direkte i databasen om der eksistere en bruger med den email og det password. Når man opretter en konto skal man indtaste password to gange, så vi sikre at brugeren har fået tastet rigtigt. Der er ikke lavet yderligere validering el. sikkerhed i forbindelse med login.

Vi har lavet vores egen DatabaseException der smider en besked op igennem systemt f.eks. fejl i forsøget på at logge ind. Vi har desværre ikke fået indsat fejlmeddelelser alle de steder det kunne være relevant, men systemet er lavet til at kunne gøre det.

Status på implementation

Opgave	Status	Uddybning
US-1	Done	
US-2	Done	
US-3	Done	
US-4	Done	
US-5	Done	
US-6	Partially done	En admin kan se alle ordre i systemet, men kan ikke se indholdet af en ordre.
US-7	Partially done	En admin kan se alle kunder i systemet, men kan ikke se en specifik kundes ordre.
US-8	Done	
US-9	Not done	Ikke implementeret
Styling	Partially done	Vi har lavet mobile first styling. Det virker, men er ikke smukt.
Figma mockup[6]	Partially done	Vi lavede mockup til 4 sider, men valgte at prioritere kodning.

[6] Bilag 6

Proces

Fra start af var vi meget fokuseret på at få anvendt kanban-metoden korrekt i vores projekt, med det formål at det altid stod klart hvad de andre i gruppen havde gang i, så vi også kunne sidde at arbejde hjemmefra. Det var vores første prioritet.

Vi gjorde også meget ud af fra start af at blive enige om systemets entiteter og databasens opbygning.

I praksis lykkedes vi rigtig godt med at anvende Kanban-metoden, men det bar også præg af at det var første gang vi gjorde det sammen som team, f.eks. mangelfulde opgavebeskrivelser el. Man lavede opgaver der ikke var på boardet. Vi samledes om vores Kanban-board hver morgen for at gennemgå dagens opgaver og tilføje nye til vores backlog. Vi forbedrede det løbende, så det blev mere klart hvad man præcis var i gang med når man havde taget en opgave.

Vi fandt ud af at det var vigtigt at gøre opgaverne meget præcise i titel og beskrivelse, for at man ikke laver overlappende ting. Det er også vigtigt at gøre opgaverne meget små - mindre end man tænker - så man når i mål med opgaverne fra dag til dag. Det er noget vi skal arbejde videre med i næste projekt.

Den første dag startede vi med at alle fire sad med hver deres opgave, men fandt hurtigt ud af at det var en fordel at arbejde 2 og 2, da den supplerende viden gør at man arbejder hurtigere.

Dette afsnit skal beskrive projektforløbet. , og hvordan kom det til at forløbe i praksis. Så beskriv:

Konklusion

Vi kan nu efter at have færdiggjort projektet konkludere, at webudvikling ikke er nemt. Der er mange elementer man skal have styr på og vi mener at man skal sikre et struktureret forløb hele vejen igennem. Eksempelvis ville det have været rart hvis vi havde styr på Navigations Diagrammet før der blev kodet. Set fra den lyse side så har vi som team rykket os utrolig meget, både indenfor programmering og projektstyring.

Vores mål har været at skabe en brugervenlig webshop for Olsker Cupcakes. Besøgende kan nu udforske det lækre udvalg af CupCakes og tilføje dem til deres indkøbskurv uden at være påkrævet at logge ind fra starten. Vi er stolte over at vi lykkedes med dette mål.

På trods af vores succes er der nogle funktioner, som vi ikke kom i mål med. Vi nåede eksempelvis ikke at komme i mål med funktionen at kunder og admin kan se ordrernes detaljer under menuen “mine ordre” for kunder og “alle ordre” for admin. Ydermere nåede vi ikke at komme helt i mål med styling. Vores prioritet var at sikre flest mulige funktionaliteter, før vi arbejdede videre med styling af webshoppen.

Vi har lært meget under denne rejse, og vi ser frem til at bygge videre på vores erfaringer og færdigheder i fremtidige projekter. Med denne CupCake webshop har vi fået forståelsen af hvordan sammenkobling mellem backend og frontend sker. Vi ser frem til at lære mere, styrke vores kompetencer og udforske nye muligheder af webudvikling.

Bilag

Link til GitHub Repository

https://github.com/babanoelk/Cupcake

Link til GitHub Kanban-board

https://github.com/users/babanoelk/projects/2/views/1

Bilag 1

https://github.com/babanoelk/Cupcake/blob/main/Cupcake/documents/ActivityDiagram.puml

Bilag 2

https://github.com/babanoelk/Cupcake/blob/main/Cupcake/documents/DomainModel.puml

Bilag 3

https://github.com/babanoelk/Cupcake/blob/main/Cupcake/documents/ClassDiagram.puml

Bilag 4

https://github.com/babanoelk/Cupcake/blob/main/Cupcake/documents/EERD.png

Bilag 5

https://github.com/babanoelk/Cupcake/blob/main/Cupcake/documents/NavigationDiagram.puml

Bilag 6

https://www.figma.com/file/SCdeLPzLmpu1DERgcl3OAq/Cupcake?type=design&node-id=0-1&mode=design