Cupcake

23/10/23 - 02/11/23

Nicolai Theis Rolin, cph-nr140@cphbusiness.dk, Sir Rolin
Nicklas Waldemar Seier Winther, cph-nw89@cphbusiness.dk, NokIkNick
Patrick Fabrin, cph-pf73@cphbusiness.dk, Odiegenx
Christian Høj, cph-ch633@cphbusiness.dk, chris4567890

2022E2Sem - Gruppe B
02/10/2023

Baggrund:

Virksomheden som skal bruge dette produkt er en cupcake-butik bosat på Bornholm, ved navn Olskers Cupcakes som der har fundet en ny opskrift som de gerne vil dele med omverdenen. Firmaet tilbyder en service hvor man kan tilpasse sine egne cupcakes, lave en bestilling og dermed hente dem i butikken.

Virksomheden søger en hjemmeside hvor:

- Deres kunder selv kan vælge en top og en bund til deres cupcakes og bestille dem.
- Kunderne skal kunne oprette en profil og logge på for at kunne se deres ordrer og dens nuværende status.
- Arbejderne skal kunne tjekke op på kundernes ordre og ændre på deres status, for at kunne give kunderne besked om potentielle ændringer.

Teknologivalg:

Teknologi:	Funktioner:
IntelliJ 2023.2.4 (Ultimate Edition).	**IDE (Integreret udviklingsmiljø).**
Github Desktop 3.3.5 (x64).	**Github-interface. Version Control.**
Postgresql 15.4.	**Database.**
pgAdmin 4 7.5.	**Database Manager.**
Javalin 5.6.1.	**Java Web Framework.**
Thymeleaf 3.1.1.	**Java Template Engine Framework**
Junit Jupiter 5.8.2.	**Unit Testing.**
HikariCP 4.0.3.	**Java Database Controller.**
Java JDK 17.0.9.	**Programmeringssprog**
Docker Desktop 4.24.2.	**Containerization Software**

Krav:

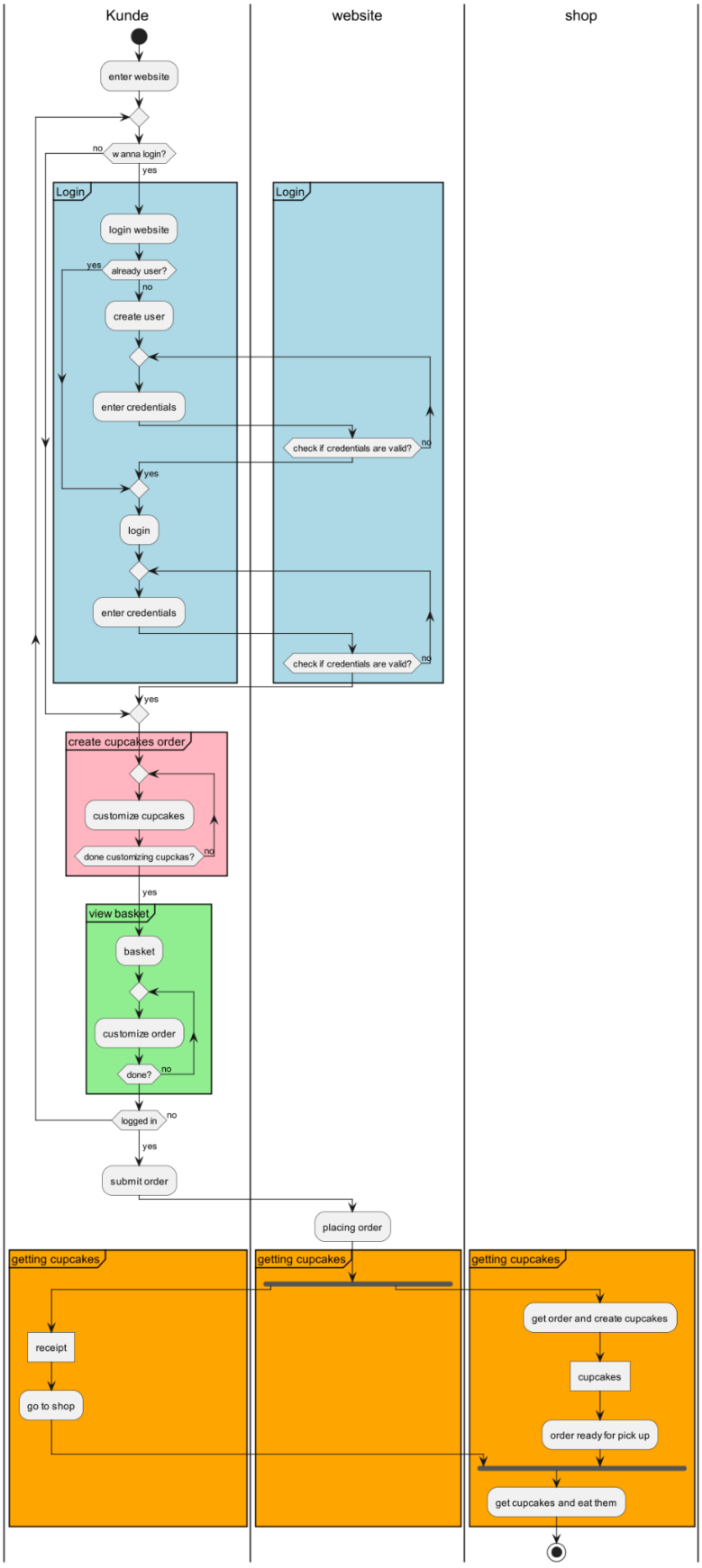
Virksomheden ønsker en hjemmeside hvor kunder kan tilpasse deres egne cupcakes via deres kollektion af toppe og bunde, og bestille dem. Der skal være en side hvorpå de ansatte kan holde et overblik over ordrerne og holde kunderne opdateret på hvor langt deres ordre er kommet, samt aflyse ordrerne hvis f.eks. en betaling ikke er gået igennem. Vores system skulle kunne gøre det nemt og brugervenligt, for en kunde at tilpasse og og bestille

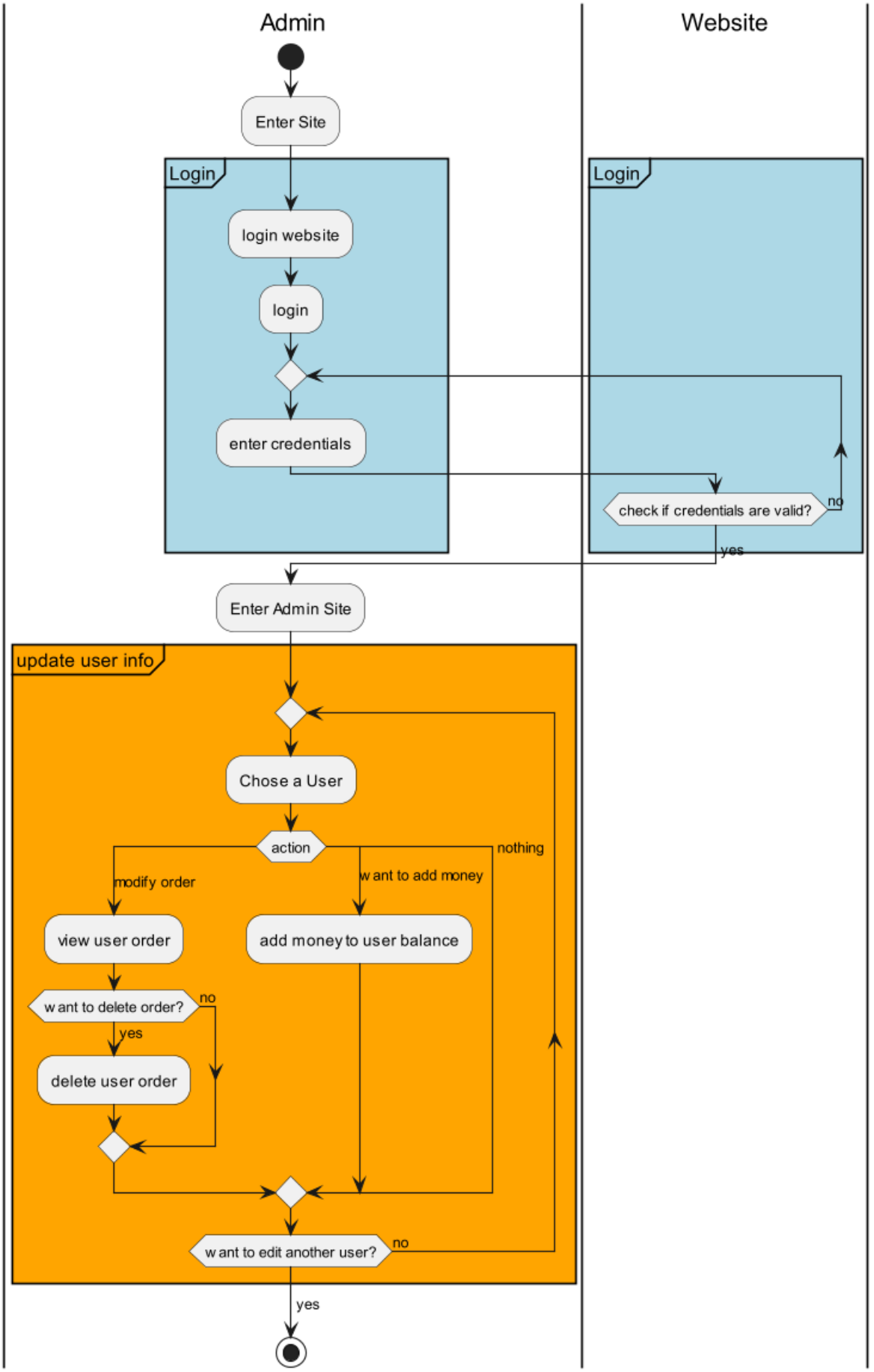
deres selvvalgte cupcakes. Det skulle også kunne hjælpe de ansatte hos Olsker Cupcake med, nemt at kunne holde et overblik over alle deres kunder og deres ordrer.

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 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, 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.

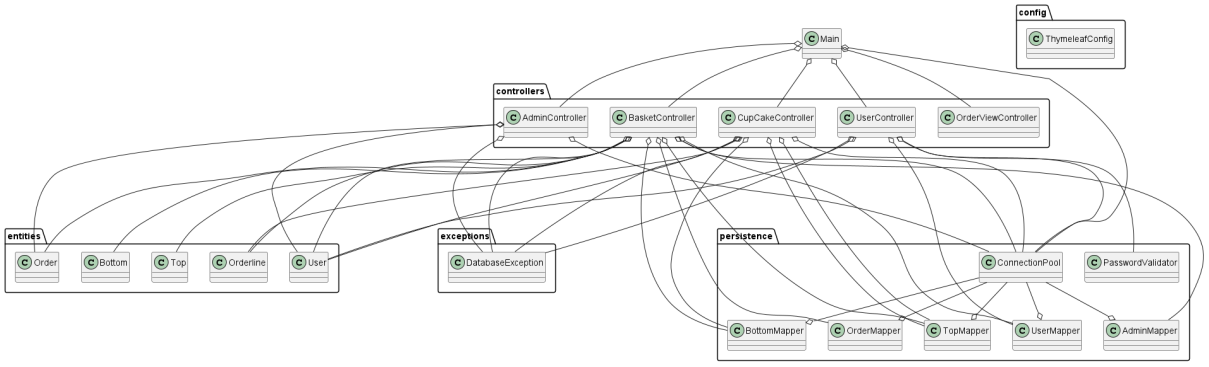
Aktivitetsdiagram:




Admin
Website
Enter Site
Login
login website
login
enter credentials
Login
check if credentials are valid?
no
yes
Enter Admin Site
update user info
Chose a User
action
nothing
modify order
want to add money
view user order
add money to user balance
want to delete order?
no
yes
delete user order
want to edit another user?
no
yes

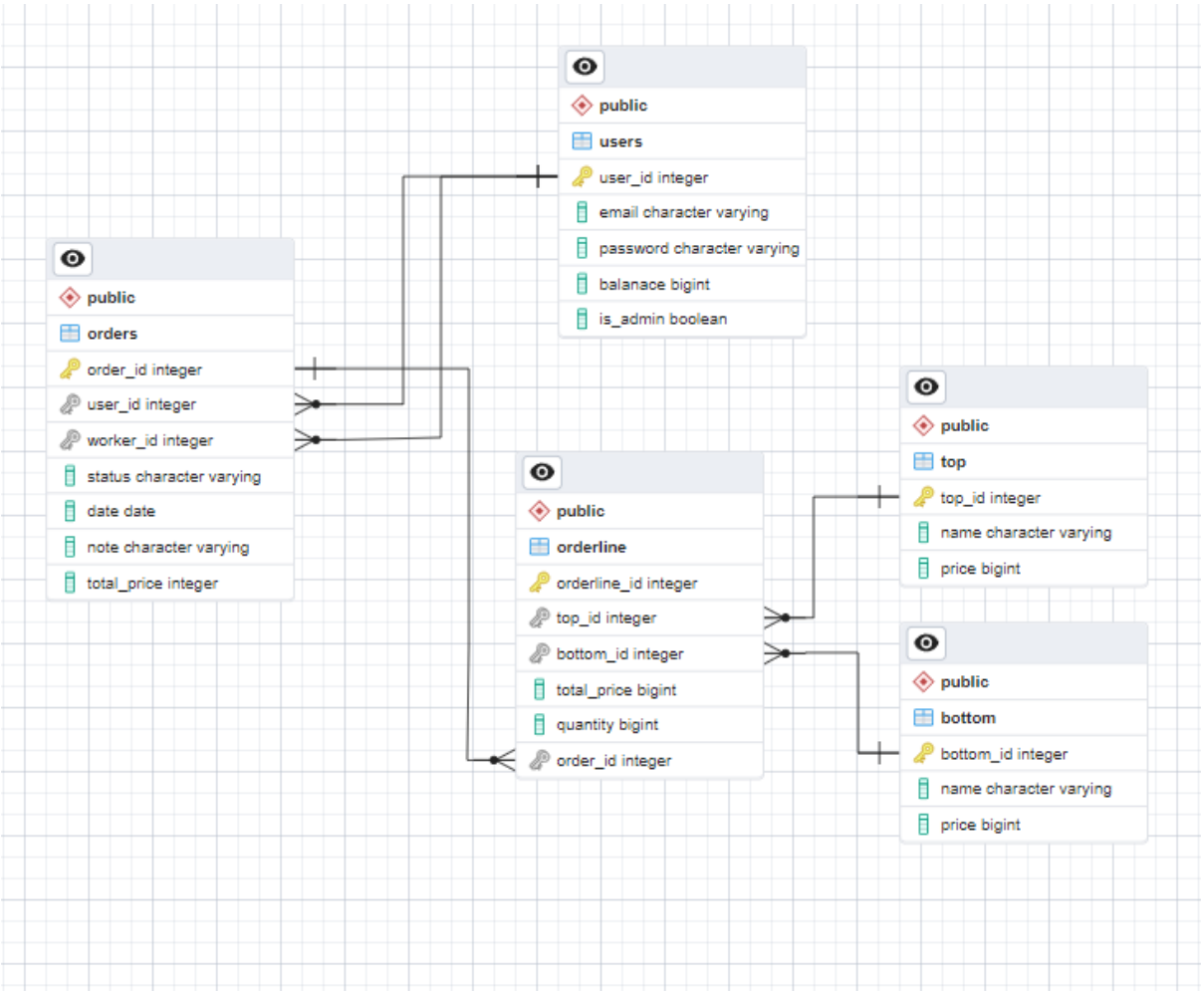
Domænemodel og ER Diagram:

Domænemodel:



hvis det er for småt er der et i docs/plantuml i projektet.

ER diagram:

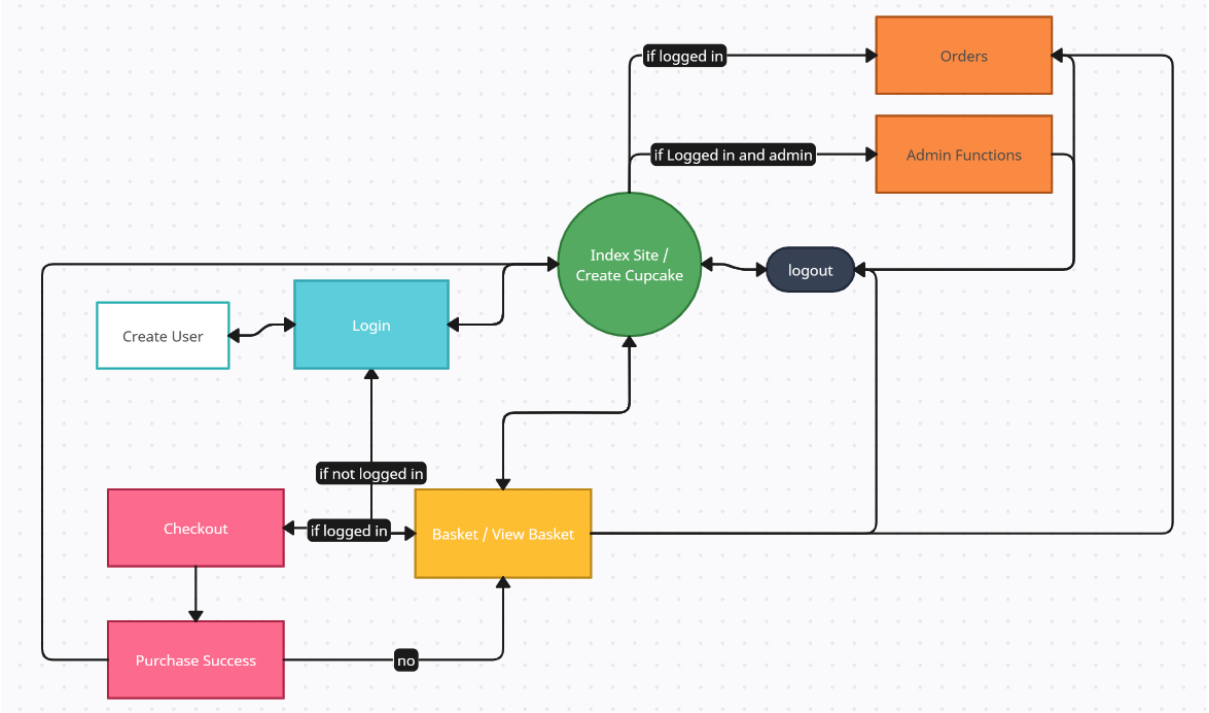


users har en en til mange relation med orders da der kan være en bruger der har mange ordrer eller kun en ordre eller slet ingen overhovedet.

orderline har en til mange relationer til tops og bottoms da man kan ha mange toppe og bottoms men en orderline kan ikke eksistere uden en top og en buttom

orderline har også en en til mange relation med orders eftersom at orders ikke rigtigt kan være en ting uden orderlines. se orderline som værende “backend” for orders.

Navigationsdiagram:



Vi har en fælles header der kan navigere alle sider (nogle af siderne er kun synlige hvis du er logget ind).
admin kan kun tilgås hvis du er logget ind som admin/har et link.
create user kan ikke blive tilgået som bruger medmindre man har et link til selve siden..

Særlige forhold:

Exceptions bliver håndteret via vores klasse DatabaseException.java, der agerer som en extension fra Exception, som vi bruger til at outputte beskeder samt fejlen den kommer med.
intet salt til enkryptering og cookien indeholder passworded burde nok laves om faktisk burde man ikke ha passworded i user entitien i det hele taget.

men den tjekker om der er et password med flere tegn på.
af brugerinput har vi password der skal indeholde et specielt tegn, et stort tegn, et tal og stort og småt og på minimum 8 karakatere og brugernavnet skal være en email så kræver @punktum for at kunne logge in.
user klassen bliver gemt i session og aktiveuser bliver gemt hvis admin, orderlines bliver gemt i session for basket. De tidligere ordrelinjer bliver gemt som en session for at kunne vise kvitteringen.

Status på implementation:

Use-cases:	Status:
UC-1: Kunden skal kunne vælge mellem en liste af toppe og bunde fra to forskellige dropdowns, og tilføje den til sin kurv form af en ordrelinje.	UC-1: Er blevet fuldt implementeret. Kunden kan trykke på de to forskellige dropdowns og vælge mellem alle de dynamisk indlæste toppe og bunde fra databasen. Derefter, bliver det tilføjet til deres kurv, som en ordrelinje.
UC-2: Kunden skal kunne oprette en profil via et registreringssystem/side, hvorpå deres adgangskode bliver valideret, for at den opnår en vis kompleksitet.	UC-2: Er blevet fuldt implementeret. Kunden kan trykke på “login” og hvis de ikke har en bruger, kan de trykke på “register” knappen for at finde registrerings siden. Derpå kan man skrive sine oplysninger, og lave en adgangskode som skal følge vores sikkerheds kriterier for at kunne oprette sig som bruger .
UC-3: Kunden skal kunne se, bestille og betale deres ordre under en “kurv” side, og derefter modtage en udskrevet kvittering.	UC-3: Er blevet fuldt implementeret. Kunden kan se hele listen af deres ordrelinjer under “kurv” siden, og kan bestille og betale ved hjælp af “submit order” knappen. Derefter, bliver der udskrevet en kvittering samt et ordrenummer.
UC-4: En administrator skal kunne indsætte penge på en valgt kundes konto direkte fra hjemmesiden igennem databasen, så de kan købe flere varer.	UC-4: Er blevet fuldt implementeret. En given admin, kan gå ind på sin “kunder” side og tildele en valgfri kunde et vilkårligt beløb og derefter indsætte det på kundens konto. Dette er gjort direkte via postgresql, men gjort sikkert med et prepared statement.
UC-4: En kunde skal kunne se sine valgte ordrelinjer i en indkøbskurv, så de kan se den samlede pris.	UC-4: Er blevet fuldt implementeret. En kunde kan se alle de ordrelinjer som de har tilføjet i deres kurv, og den samlede pris bliver beregnet og vist nederst på ordren, på kurv siden.

UC-5: Når en bruger er logget på, skal de kunne se deres e-mail på hver side..	UC-5: Er blevet fuldt implementeret. Den bruger som er logget ind kan se den e-mail som de er registreret og logget ind med, og de kan også trykke på den for at logge ud.
UC-6: Administrator kan se alle ordrer i systemet, så de kan se hvad der er blevet bestilt.	UC-6: Er blevet fuldt implementeret. Man kan tilgå siden “alle ordrer", hvis man er logget ind som administrator. Siden vil ikke dukke op hvis man er logget ind som normal bruger. Inde på siden kan man tilgå alle ordrer på systemet og se alle kunder.
UC-7: Som administrator kan man se alle kunder i systemet og deres ordrer, så man kan følge op på ordrer og holde styr på kunderne.	UC-7: Er blevet fuldt implementeret. Så længe man er logget ind som administrator, kan man tilgå siden “alle ordrer”, derpå kan man se alle kunder og deres ordrer, samt deres ordre status.
UC-8: Som kunde kan de fjerne en ordrelinje fra deres indkøbskurv, så de kan justere deres ordre.	UC-8: Er blevet fuldt implementeret. Som kunde kan man justere ens kurv, ved at fjerne ordrelinjer inde på “kurv”-siden. Når man fjerner en ordrelinje, vil den samlede pris automatisk opdatere.
UC-9: Som administrator, kan man fjerne en ordre direkte på hjemmesiden, hvis en ordre nu skulle være ugyldig, så som ikke betalt eller lign.	UC-9: Er blevet fuldt implementeret: Som administrator, kan man tilgå siden “alle ordre” som viser alle kunder og deres ordre, samt deres status. Deri, kan man slette ordre, på “delete” knappen, hvis nu, at der skulle være noget galt.
UC-10: Hver gang, at butikken får en ny mulig top eller bund, vil hjemmesiden automatisk blive opdateret med den mulighed i dropdown menuerne.	UC-10: Er blevet fuldt implementeret. Hvis butikken får en ny mulig top eller bund, kan de tilføje det til databasen. Hjemmesiden vil automatisk generere mulighederne i dropdown-menuerne, hvis de nye toppe og bunde er blevet tilføjet til databasen.
UC-11: Som administrator, kan man ændre status på enhver kundes ordre ved hjælp af hjemmesiden.	UC-11: Ikke implementeret. Man kan vælge en ordre ved hjælp af “alle ordre” knappen som administrator og se alle ordre, men man kan ikke ændre status på ordren.
UC-11: Som administrator, kan man tilgå enhver ordre og tildele sig selv eller andre arbejdere en ordre, som de skal arbejde på, ved hjælp af hjemmesiden.	UC-11: Ikke implementeret. Man kan vælge en specifik ordre ved hjælp af “alle ordre” knappen som administrator, men man kan ikke tildele sig selv eller andre opgaven at bage cupcakesne.

UC-12: Som administrator, kan man trykke på en knap eller dropdown for at filtrere mellem alle ordrer på deres status, for nemmere at kunne danne sig et overblik.	UC-12: Ikke implementeret. Man kan tilgå siden “alle ordrer” som administrator, derpå kan man kan se de specifikke ordrer, men man kan ikke filtrere dem.
UC-13: Hjemmesiden skal være tilgængelig og fungerende på samtlige devices, herunder: Computere og Mobile enheder.	UC-13: Delvist implementeret. Hjemmesiden kan tilgås og er fuldt fungerende på en computer og laptop. Hjemmesiden kan ikke vises korrekt på mindre skærme så som mobile enheder.
UC-14: Hjemmesiden skal have en administrator side, som kun kan tilgås hvis man er administrator. Siden skal være usynligt for normale brugere.	UC-14: Delvist implementeret. Hjemmesiden har en fungerende administratorside, kaldet “alle ordrer”. Man kan tilgå den ved hjælp af knappen af samme navn, og den knap dukker kun op for brugere der gælder som administrator. Der er en lille fejl i, at siden stadigvæk kan tilgås hvis man har et link, trods, at man ikke er admin.

Video af det færdige produkt:

https://youtu.be/fQpHWcXz8-M

Proces:

Vores gruppe kæmper ofte med planlæggelsesfasen og derfor ville der gerne fokuseres mere på den del af udviklingsfasen. Struktur og planlægning var hovedidéen da projektet begyndte, og derfor blev der skabt et kanban-board via Githubs projektfunktion. Planen var at holde sig til kanban-boardet, og tage et user-case af gangen. Hver gang, at der ønskes at tilføje en ny feature der ikke var planlagt, ville den blive tilføjet til kanban-boardet og kun arbejdet på, hvis andet var lavet, eller hvis det var absolut nødvendigt. Et andet fokus som gruppen havde var, at sørge for, at lave en ny branch pr. nye implementation, på den måde, kunne vi mindske risikoen for, at overskrive vores minimum viable product.

Det skal siges, at holde sig til kanban-boardet gik rigtigt godt, og næsten hvert eneste user-case kom i mål. Der blev kun arbejdet på hvad det krævede, at få et minumum viable product først, og derefter blev der tilføjet en masse “nice-to-have” features.

En ting der skal tages med til næste projekt er, at planlæggelse for klasser og deres attributter skal sættes i sten før produktion, så alle er enige om, hvilke attributter og metoder en klasse har, så det er nemmere at arbejde med, selv hvis man ikke har klasserne endnu. Gruppen skal også arbejde på, at være mere enige i, hvordan gruppen har planlagt at skulle

implementere forskellige features, da der til tider var en misforståelse for hvad en feature skulle gøre og hvordan det skulle opnås.

Gruppen fik lavet interaktion med databasen via prepared statements meget tidligt i produktionsforløbet, og det gjorde det nemt at lave resten af klasserne, da alt data man skulle bruge nemt kunne hentes. Det vil helt klart huskes til kommende projekter. Det var også nyttigt, at gruppemedlemmet der stod for database kommunikation og gruppemedlemmet der stod for mapping af den data der kom fra databasen, havde snakket samme om hvad hinanden havde brug for, så der ikke var behov for, at gå tilbage til database controllers for, at justere eller ændre noget.

Til næste projekt, ville det også være oplagt at arbejde lidt på gruppens forventninger og kommunikation. Der har været lidt misforståelse for, hvor og hvornår gruppen har skulle mødes eller hvem der har skulle stå for hvad i projektet. Der skal også arbejdes på tidsfordeling af arbejde, da nogle ting måske har taget lidt længere tid end behøvet.

Produktiviteten var høj og kode kvaliteten har været udmærket, med minimale bugs og fejl. Og med det sagt, vil gruppen vurdere projektet som værende succesfuld med en masse at tage med til næste projekt.