

Pizzeria

Informatikk Prosjektarbeid 1, laget av gruppe 17: Haakon Sønsteby, Torunn Arnesdatter Sæther og Sindre Svendsrud

Innholdsfortegnelse

Innledning.....	2
Målgruppe	2
Brukermanual for programmet	3
Kommentering av databasen	11
Kodeforklaring:	13
Databasepakka:	13
ObjektTyper:.....	14
testFiler:.....	14
Kart forklaring:.....	14
GUI forklaring:	14
JARS	16
Parprogrammering	16
Oppdatering mellom maskiner.....	16
Brukertesting	17
Utvidelser	17
Avslutning	17
Appendiks A:.....	18
Klassediagram.....	19
Klassediagram2.....	20
Use case diagram.....	21
ER- diagram	22
Appendiks B:.....	23
Møtereferater	23
Foreløbig plan via Scrum:	27
Kommentarer etter midtveispresentasjon.....	28
Brukeresting:	28
Appendiks C:.....	30
Risikoanalyse	30
Timeliste:	30
Backlog	33
Sprint1:	34
Sprint2:	35
Sprint3:	35

Sprint4:	36
Sprint5:	36
GUIskisser	36

Innledning

Vi har fått en prosjektoppgave hvor vi skal lage et system for å motta bestillinger for en valgfri restaurant. Bestillingene skal mottas over telefon og legges inn i systemet. Systemet vi har laget legger også inn kundeinformasjon samt at det inneholder brukergrensesnitt for kokken og leveringsbudet. Programmet inneholder også en admin side hvor man har muligheten for å administrere kunder, bestillinger, meny og endre fraktpris. Vi har valgt å lage systemet for en pizzarestaurant.

For å lage programmet har vi gjort en del antagelser og forutsetninger underveis. Vi har valgt at restauranten bare skal kunne lage pizzaene og levere disse (evt. hentes av kunde) og at det derfor ikke skal være noe spisested. Videre for å få tilgang til systemet må man ha internettilgang til Pcen som skal kjøre programmet, dette er for å få kontakt med databasen hvor all informasjon ligger lagret. Pcen må også ha en minimum oppløsning på WXGA (1366 x 768) for å få vist alle elementene i programmet. For å få hentet ut kart over kundens adresse (evt. leveringsadresse) er man avhengig av at kunden oppgir riktig veinavn og nummer, da programmet vårt ikke har noen måte å sjekke om veinavn og nummer faktisk finnes på oppgitte postnummer. Prisene som er oppgitt i programmet er oppgitt med og uten moms, vi har da satt momsen til å være på 14 %. Restauranten har en åpningstid mellom 14-24, og det er i dette tidsrommet man kan legge inn bestillinger. Man har bare mulighet til å levere pizzaene hver halvtime, dette er for å effektivisere bestillingsprosessen.

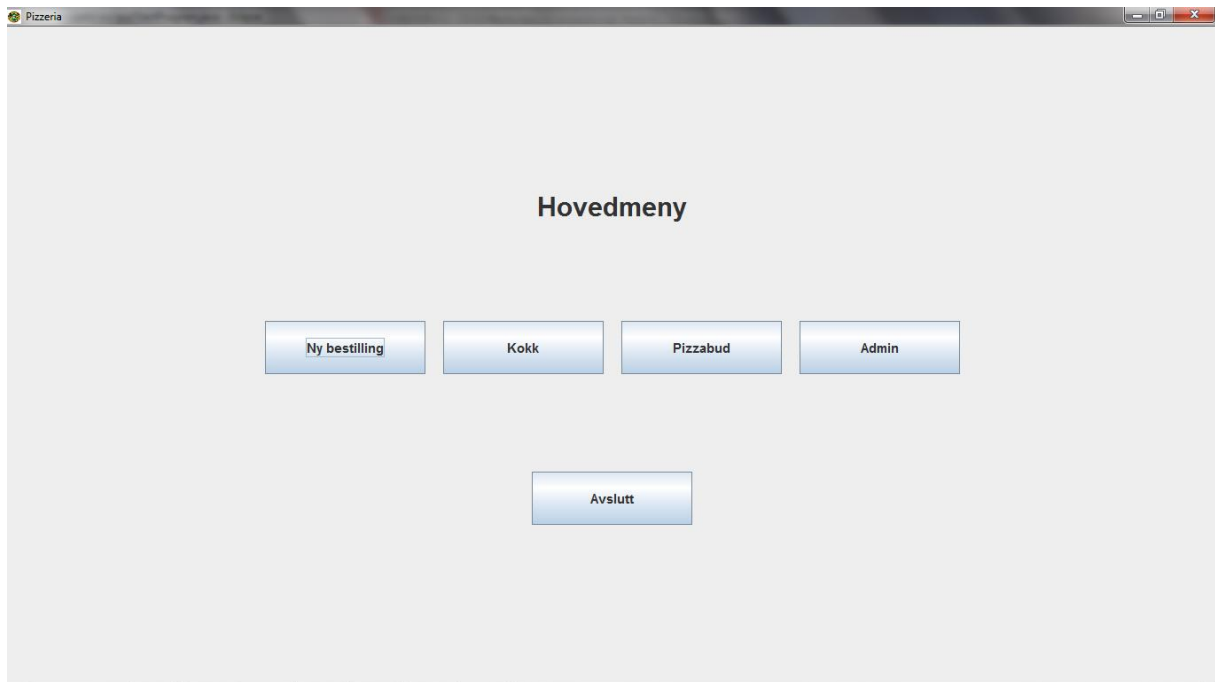
Målgruppe

Som det står i kravspesifikasjonen er målgruppen de ansatte i en restaurant. En restaurant har som kjent flere arbeidsgrupper, fra servitører, til kokk og leveringsbud, i tillegg til aldersforskjell på de forskjellige yrkene. Dette gjør det vanskelig å designe programmet på den mest brukervennlige måten ettersom systemet skal brukes av så mange forskjellige personer. For å få et best mulig program er det viktig å avgrense målgruppen så mye som mulig. Dette for å slippe å ta hensyn til f.eks. barn i 10-15 års alderen som gjerne vil ha farger og animasjoner eller eldre som alltid ønsker større skrift på knappene.

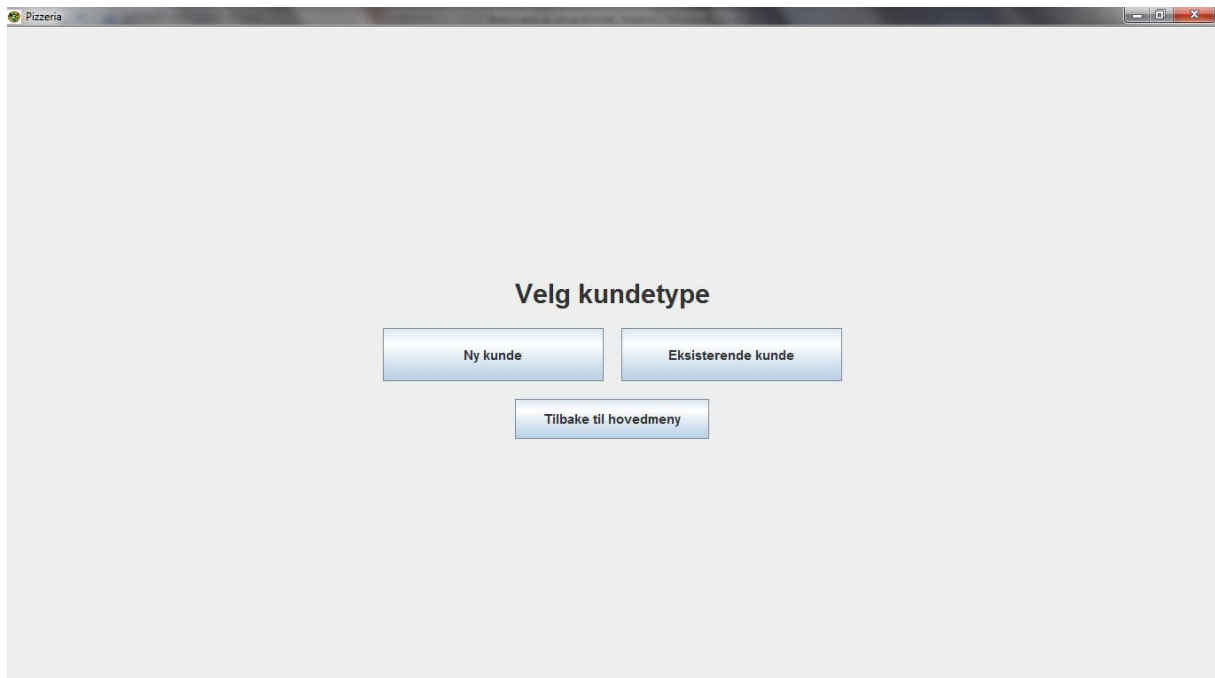
I vårt program har vi valgt å la målgruppen være for personer mellom 20-40 år, mao et stort spenn. Dette ikke er optimalt, men vi har gjort dette for at alle skal få en like god opplevelse av programmet. På denne måten får ikke noen personer en veldig positiv opplevelse med programmet, mens andre veldig negativ opplevelse. Ulempen er selvfølgelig at programmet ikke blir så brukervennlig som det kanskje kunne blitt dersom det var en mer spesifisert målgruppe.

Brukermanual for programmet

Vi har valgt å dele programmet vårt i 4 hoveddeler: Ny bestilling, Kokk, Pizzabud og en Admin side.



For å legge inn en ny bestilling har vi en knapp "Ny bestilling" i hovedmenyen. Hvis man går inn her har man valget å legge til en "ny kunde" eller "eksisterende kunde".



Hvis man velger eksisterende kunde, så søker man kunden opp ved hjelp av telefonnummeret eller navn og kan her enkelt endre alle feltene til kunden, eller slette kunden om det er ønskelig. Hvis man velger ny kunde, så setter man alle feltene til den nye kunden og går videre til bestillingen.

Finn eksisterende kunde

Telefonnummer Fornavn

Navn Etternavn

Adresse

Kunder

Postnummer HØNEFOSS

Telefon

Legg til ny kunde

Fornavn

Etternavn

Adresse

Postnummer

Telefonnummer

Det er to tabeller som viser menyen og bestillingen underveis. Man velger de pizzaene man vil ha fra menyen ved å enkelt trykke på dem, så kommer pizzaene opp på bestillingen. Hvis man vil kan man legge inn en kommentar, hvis man f.eks. ikke vil ha en ingrediens som er på pizzaen(e), er dette en enkel måte å si fra til kokken på.

Pizzeria

Kunde

Navn: h jf Telefon: 77777777

Meny

Nummer	Navn	Pris	Beskrivelse
195	Pizza	800	God
202	kdefj	400	jdsh
205	Sindre	600	God
206	PizzaBest	700	Stor
207	Pepperoni	300	Uten løk
210	Vegatar	100	Uten kjøtt
212	Dessertpizza	100	Med is
213	nummer en	56	tomat og ost
214	ImpressME	500	Digg
215	Pizza10	210	Ny pizza
216	Pizza10	210	Ny pizza
217	Pizza10	210	Ny pizza
218	Pizza11	211	Ny pizza
219	Pizza12	210	Ny pizza
220	Pizza13	210	Ny pizza
222	Pizza15	210	Ny pizza
223	Kokkens spesial	800	fantasSStsk
224	TestRett	700	besk
225	HAMBURGER	1337	ikke pizza
227	Pizza42	200	ikke god
228	plzzaa	700	ja
229	Karis spesial	89	pizzabunn med cream fresh, kylling, pesto og roc...
230	navn	377	olo
231	la	6	mei

Bestilling

Nummer	Navn	Pris	Beskrivelse	Antall
212	Dessertpizza	100	Med is	1
214	ImpressME	500	Digg	1

Fjern

Pris uten moms: 600,00 kr.

Pris med moms: 684,00 kr.

Kommentarer
[Vil ikke ha mais på pizzaen]

OK Tilbake til hovedmeny

I tillegg velger man om bestillingen skal hentes, eller leveres. Hvis den ønskes levert, må man oppgi en leveringsadresse eller bruke kundens hjemmehadresse. Man må også oppgi når bestillingen skal leveres. Klokketallet er automatisk satt til 14:00 som er åpningstiden til restauranten, mens datoen, måneden og året er default satt til systemdatoen så man slipper å endre på dette om man skal ha en pizza på samme dag som man legger inn bestillingen (men kan endres om ønskelig).

Pizzeria

Leveringsdetaljer

Kunde

Navn: h jf

Adresse: ui

9001 TROMSØ

Telefonnummer: 77777777

☐ Hentes

☒ Leveres

☒ Kundens adresse

☐ Leveringsadresse: []

Postnummer: []

Leveringstidspunkt: 15:30 18 November 2010

Bestilte retter

Retter:

Retter:

Dessertpizza x 1 100kr.

ImpressME x 1 500kr.

Pris uten moms: 600,00 kr.

Pris med moms: 684,00 kr.

Fraktpris: 0,00 kr.

Totalpris: 684,00 kr.

Endre retter OK Tilbake til hovedmeny

Når man har valgt leveringsdetaljene kommer man til et vindu hvor kvitteringen står, her kan man velge å printe den ut.

Kvittering

Kunde: Ole Nordmann
 Adresse: Høgskoleringen 7
 7030 TRONDHEIM
 Telefonnummer: 12345678

Retter:
 HomeMade x 1 250kr.

Pris uten moms: 250,00kr.
 Pris med moms: 285,00kr.
 Fraktpris: 50,00kr.
 Totalpris: 335,00kr.

Kommentar:

Leveringstidspunkt: 14:00 19.11.2010
 Leveringsadresse: Høgskoleringen 7
 7030 TRONDHEIM

OK Skriv ut

For at kokken skal ha oversikt over bestillingene, så har vi laget en side "Kokk" som viser en liste over de bestillingene som ikke har blitt laget. Her vises bestillingene med pizza, antall, kommentar og leveringstidspunktet. Når kokken har laget ferdig en bestilling kan han velge denne bestillingen og trykke "ferdiglaget", bestillingen vil da være ute av denne delen av systemet.

Ulagede bestillinger

BestillingsID	Retter	Kommentar	Hentes/Leveres	Leveringstidspunkt
1	Pizza, Pizza, Pizza, Pizza, Pizza		Leveres	14:30 16.11.2010
3	Test, PizzaBest		Leveres	16:00 16.11.2010
6	Pepperoni		Leveres	16:00 16.11.2010
5	ImpressME, ImpressME, ImpressM...		Leveres	16:00 16.11.2010
7			Leveres	17:00 16.11.2010
14			Leveres	17:00 16.11.2010
16	Pizza10, Pizza10		Leveres	17:00 16.11.2010
22			Leveres	17:30 16.11.2010
9			Leveres	18:00 16.11.2010
8			Leveres	18:30 16.11.2010
41	Pizza10		Leveres	14:00 17.11.2010
40	Pepperoni, Pizza11		Leveres	15:00 17.11.2010
15			Leveres	14:00 18.11.2010
28	Kokkens spesial	null	Leveres	14:00 19.11.2010
30	Kokkens spesial		Leveres	14:00 19.11.2010
32	Pizza12	null	Leveres	14:00 19.11.2010
34	Pizza13, nummer en, Karis spesial...	null	Leveres	14:00 19.11.2010
38	Pizza10, Pizza12, Kokkens spesial		Leveres	14:00 19.11.2010
35	Dessertpizza	null	Leveres	17:00 19.11.2010
23	Pizza10		Leveres	14:00 20.11.2010
26	Pizza12		Leveres	14:00 20.11.2010
29	Pizza12		Leveres	14:00 20.11.2010
31	HAMBURGER		Leveres	14:00 20.11.2010
33	Pizza10		Leveres	14:00 20.11.2010

Ferdiglaget Tilbake til hovedmeny

Rettene:

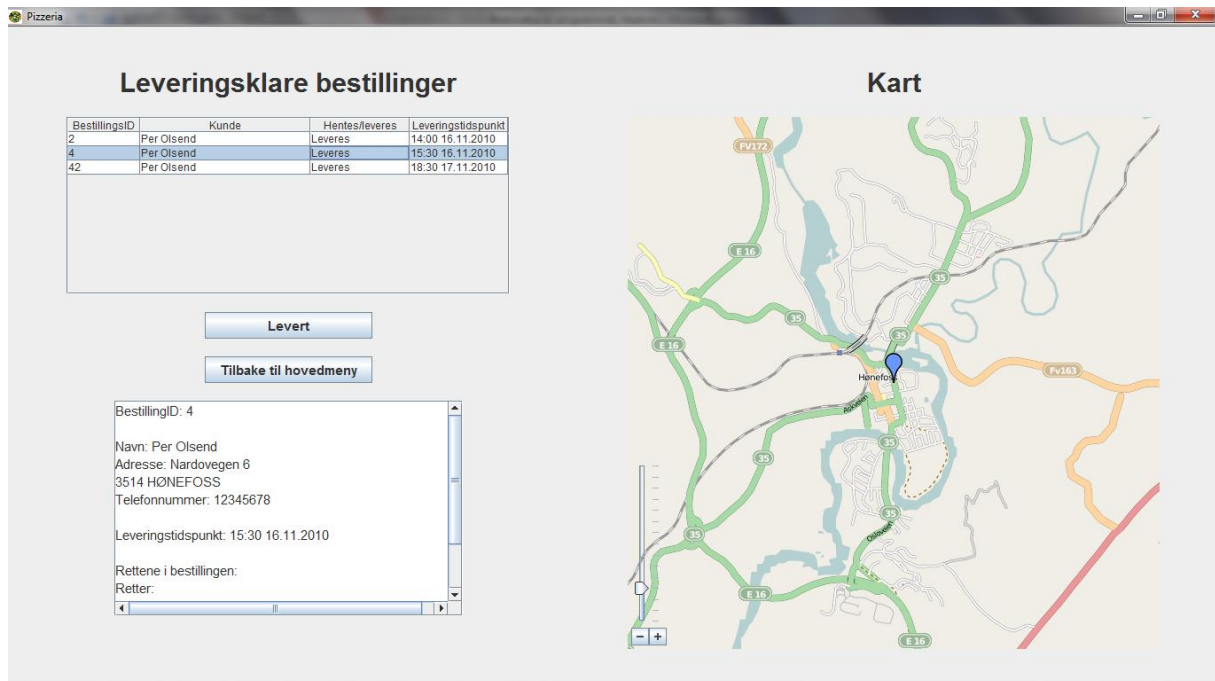
BestillingID: 6

Rettene i denne bestillingen:
 Rett1: Pepperoni

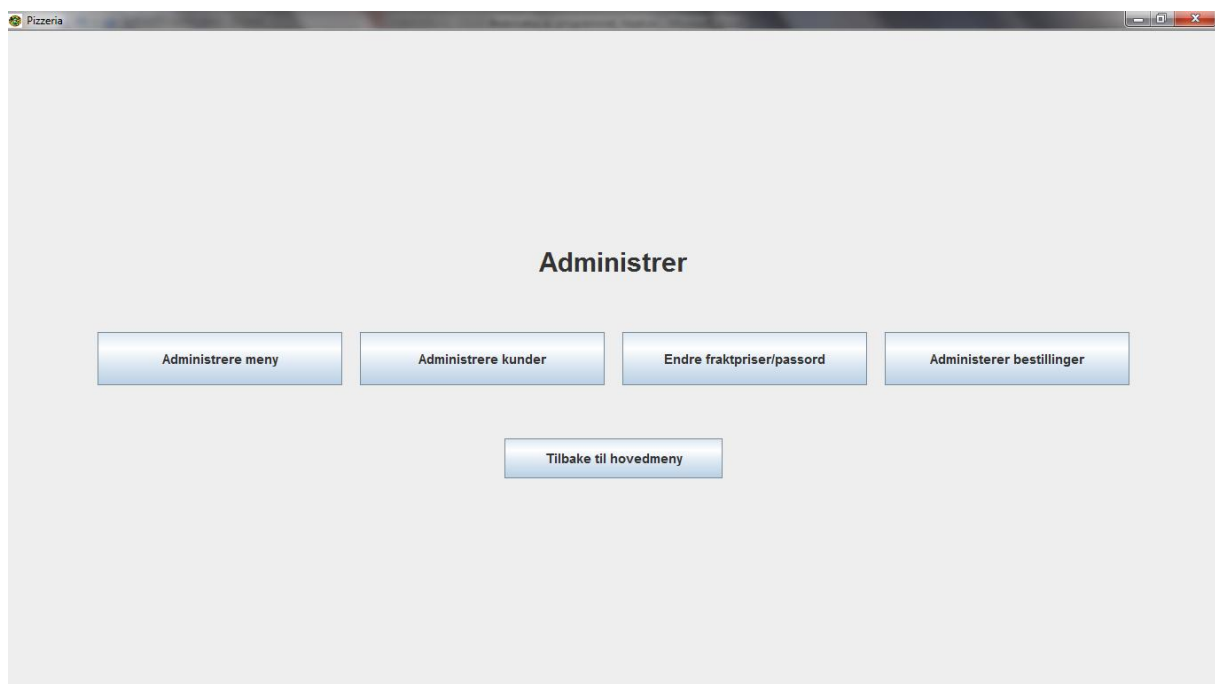
Kommentar:

Leveringsbudet må ha en oversikt over hvilke bestillinger som er laget, men ikke levert. Han skal i tillegg ha mulighet for å hente et kart over hvor leveringsadressen befinner seg. Leveringsbudet får derfor opp en liste med leveringsklare bestillinger, med all informasjonen som trengs for å levere bestillingen. Ved å merke en av bestillingene, ser man leveringsadressen på kartet til høyre for lista.

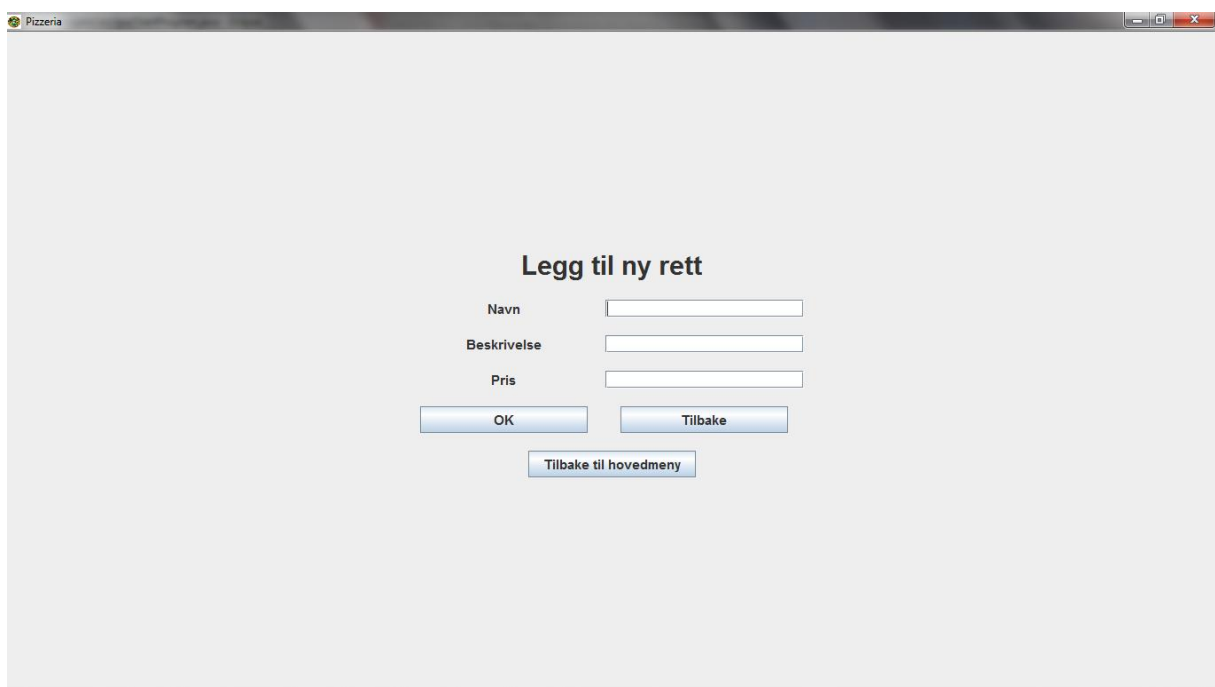
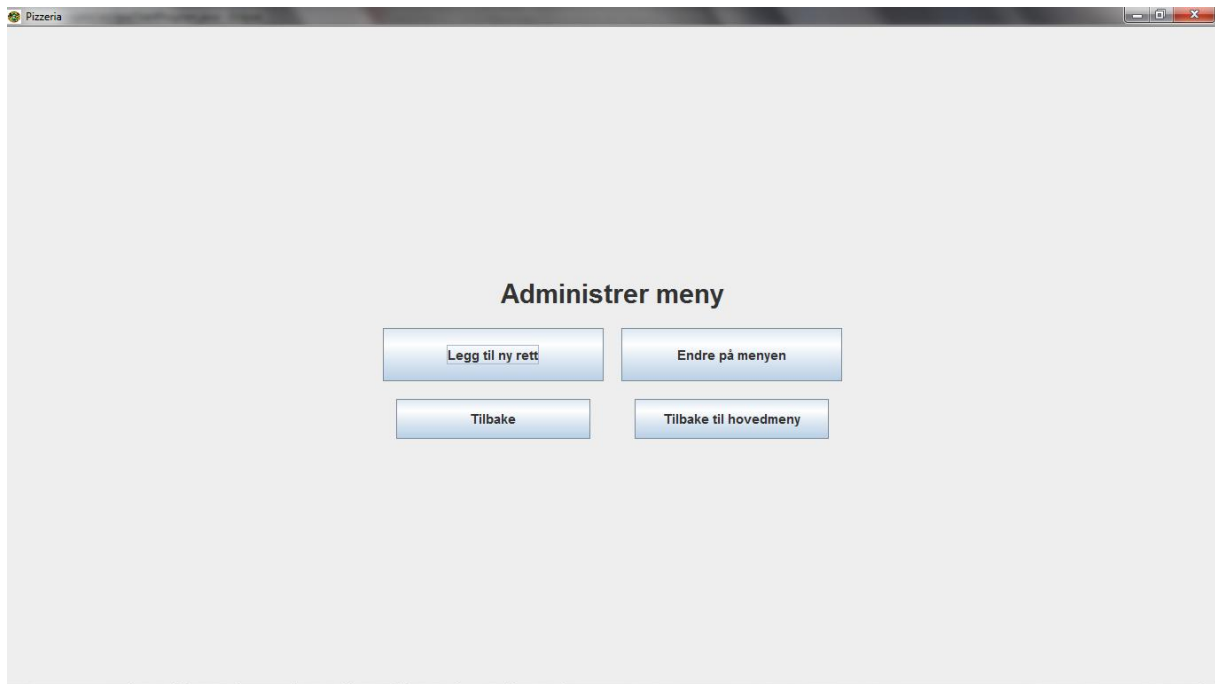
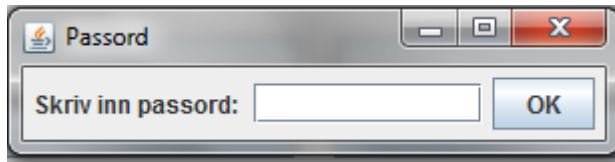
Kartet har mulighet til å zoomes inn/ut og dras i de retninger man ønsker. Når en bestilling er levert merker man bestillingen og trykker lever. Bestillingen blir da borte fra denne lista.



Administratorsiden er delt opp i fire deler: Adminstrer meny, Administrer kunder, endre fraktpriser/passord og administrer bestillinger. Administratorsiden er ment for de administrative oppgavene, som i hvertfall ikke kokk og leveringsbudet skal ha tilgang til. Vi tenkte på å ha et passord som beskyttet hele administratorpanelet, men kom fram til at de ansatte burde ha tilgang til "Administrer bestillinger" og "Administrer kunder". Det er derfor kun "Administrer meny" og "Endre fraktpriser/passord", som krever et passord.



Under “administrer meny” har man mulighet til å legge til en rett i menyen, fjerne en rett i menyen, samt endre en eksisterende rett. I stedet for å slette rett, blir den bare satt slik at den ikke er synlig. På denne måten unngår vi problemer hvis vi ønsker å hente fram bestillinger med retter som ikke er på menyen lengre.



Nummer	Navn	Beskrivelse	Pris	På menyen?
195	Pizza	God	800	ja
202	Kdshj	jdsh	400	ja
205	Sindre	God	600	ja
206	PizzaBest	Stor	700	ja
207	Pepperoni	Uten løk	300	ja
210	Vegetar	uten kjøtt	100	ja
212	Dessertpizza	Med is	100	ja
213	nummer en	tomat og ost	56	ja
214	ImpressME	Digg	500	ja
215	Pizza10	Ny pizza	210	ja
216	Pizza10	Ny pizza	210	ja
217	Pizza10	Ny pizza	210	ja
218	Pizza11	Ny pizza	211	ja
219	Pizza12	Ny pizza	210	ja
220	Pizza13	Ny pizza	210	ja
222	Pizza15	Ny pizza	210	ja
223	Kokkens spesial	fantasSStsk	800	ja
224	TestRett	besk	700	ja
225	HAMBURGER	ikke pizza	1337	ja
227	Pizza42	ikke god	200	ja
228	pizzaa	ja	700	ja
229	Karis spesial	pizzabunn med cream fresh, kylling, pesto o...	89	ja
230	navn	olo	377	ja
231	ja	nel	6	ja

Navn:

Beskrivelse:

Pris:

☒ På menyen

Under “administrer kunder” har man mulighet til å endre personlig data på en eksisterende kunde, slette en kunde, eller legge til en ny kunde. Dette er samme panelet som du kommer til dersom du går via hovedmenyen og “Ny bestilling”, men her har du i tillegg muligheten til å slette kunde, noe du ikke hadde mulighet til dersom du kom andre veien. Du har i tillegg ikke mulighet til å gå videre (for å opprett en ny bestilling), det er kun opprette/endre/slette kunde oppgaver som er mulig.

Velg kundetype

The screenshot shows a web application window titled 'Pizzeria'. The main heading is 'Finn eksisterende kunde'. Below the heading, there are two columns of input fields. The left column contains 'Telefonnummer' and 'Navn'. The right column contains 'Fornavn', 'Etternavn', 'Adresse', 'Postnummer', and 'Telefon'. A blue 'Søk' button is positioned below the 'Navn' field. Below the 'Søk' button is a table with the heading 'Kunder' and an empty body. To the right of the table are two buttons: 'Endre' and 'Slette'. At the bottom of the form are two buttons: 'Tilbake' and 'Tilbake til hovedmeny'.

The screenshot shows a web application window titled 'Pizzeria'. The main heading is 'Legg til ny kunde'. Below the heading, there are five input fields arranged vertically: 'Fornavn', 'Etternavn', 'Adresse', 'Postnummer', and 'Telefonnummer'. Below these fields are three buttons: 'OK', 'Tilbake', and 'Tilbake til hovedmeny'.

Under "Endre fraktpriser/passord" har man mulighet til å endre fraktprisen, samt endre grensen som sier om en bestilling skal få gratis frakt. Her har man også mulighet til å se den nåværende fraktpris, og den nåværende grensen for fri frakt. I tillegg kan man endre passord om man oppgir det gamle passordet.

Endre priser for frakt eller passord

Fraktpriis Fraktpriis: 75

Grense for fri frakt Grense for fri frakt: 500

Passord

Nytt passord:

Gjenta nytt passord

Under “administrer bestilling” kan man endre eller slette bestillinger: Her får de ansatte opp de bestillingene som ikke er levert (men de kan være laget av kokken), slik at dersom kunden ønsker å endre en bestilling, kan de ansatte gå inn her, for så å endre alt som har med bestillingen å gjøre. Da åpnes samme panel som når en ny bestilling opprettes.

Bestillinger som ikke er levert

ID	Kunde	Retter	Leveringstidspunkt
2	Per Olsend	Hakons, Hakons, Hakons, Hakons, ...	14:00 16.11.2010
1	Per Olsend	Pizza, Pizza, Pizza, Pizza	14:30 16.11.2010
4	Per Olsend	Pizza13, kdsfj, kdsfj	15:30 16.11.2010
3	Per Olsend	Test, PizzaBest	16:00 16.11.2010
6	Per Olsend	Pepperoni	16:00 16.11.2010
5	Per Olsend	ImpressME, ImpressME, ImpressME, I...	16:00 16.11.2010
7	Per Olsend		17:00 16.11.2010
14	Per Olsend		17:00 16.11.2010
16	Per Olsend	Pizza10, Pizza10	17:00 16.11.2010
22	Per Olsend		17:30 16.11.2010
9	Per Olsend		18:00 16.11.2010
8	Per Olsend		18:30 16.11.2010
41	Per Olsend	Pizza10	14:00 17.11.2010
40	Per Olsend	Pepperoni, Pizza11	15:00 17.11.2010
42	Per Olsend	Kokkens spesial	18:30 17.11.2010
15	Per Olsend		14:00 18.11.2010
28	Per Olsend	Kokkens spesial	14:00 19.11.2010
30	Per Olsend	Kokkens spesial	14:00 19.11.2010
32	Per Olsend	Pizza12	14:00 19.11.2010
34	Per Olsend	Pizza13, nummer en, Karis spesial, e,oe	14:00 19.11.2010
38	Per Olsend	Pizza10, Pizza12, Kokkens spesial	14:00 19.11.2010
35	Per Olsend	Dessertpizza	17:00 19.11.2010
23	Per Olsend	Pizza10	14:00 20.11.2010
26	Per Olsend	Pizza12	14:00 20.11.2010

Kommentering av databasen

Databasen vår består av totalt 6 entiteter. Se vedlagt grafiske oversikt.

Entiteten *Kunde* inneholder attributtene kundeNr, fornavn, etternavn, tlf, adresse, og postNr. Som primærnøkkel har vi valgt å lage en kundeID som genereres automatisk når kunde legges inn i databasen. Vi vurderte tlf som primærnøkkel, men kom fram til at en kunde burde kunne endre sitt telefonnr og at det på bakgrunn av dette ble mest oversiktlig å ha en egen primærnøkkel. Tlf nr brukes for å søke opp kunden dersom den skal legge inn bestilling. Dersom de har glemt dette, er det også mulig å søke ved hjelp av fornavn, etternavn eller hele navnet. Postnr har en referanse til post-entiteten og er derfor en fremmednøkkel.

Entiteten *Bestilling*, inneholder attributtene bestillingsNr, KundeNr, PostNr, leveringstidspunkt, leveringsadresse, totalPris, annet, levert og laget. Primærnøkkel er en bestillingsID, ettersom ingen av de andre attributtene er unike. Kundenr er kundereferansen, som sier hvilken kunde som har opprettet bestillingen. Noe vi merket etterhvert var at vi i utgangspunktet hadde gjort en for dårlig jobb med å tegne arkitekturen. Dette medførte at vi møtte på et problem når vi ønsket å ha flere enn én rett i en bestilling. Vi måtte nå tenke på nytt og kom fram til at vi måtte ha en egen entitet som vi kalte *OrdreLinje*. Nå legger vi da altså til en ordrelinje for hver rett i ordrelinjeentiteten, hvor hver ordrelinje inneholder en referanse til hvilken bestilling den tilhører. Det ble selvfølgelig andre ting som f.eks. totalprisen til en bestilling som da måtte skrives om, ettersom man da må inn i ordrelinja, finne prisen til den retten som ligger der for så å oppdatere den bestillingen som har den ordrelinja.

Entiteten *OrdreLinje* inneholder attributtene linje, OrdreID og rettID. Linje er primærnøkkel. Ordrelinje har to funksjoner. Den første er å linke sammen aktuelle rett med riktig bestilling og den andre er å linke sammen riktig rett fra entiteten rett.

Entiteten *Rett* inneholder attributter som navn, beskrivelse, pris etc. Som primærnøkkel valgte vi en egen rettID ettersom flere retter kan ha samme navn. RettID'en genereres automatisk av databasen når rett legges inn. Rett har en attributt, synlighet, som sier om retten skal være synlig på menyen eller ikke. Grunnen til dette er at gamle bestillinger, kan inneholde disse rettene, og dermed må disse rettene fortsatt finnes i systemet. En rett blir altså ikke slettet, bare satt som usynlig. På admin menyen, under administrer meny får du oversikt over hvilke retter som er synlig og hvilke som ikke er det.

Entiteten *Post* inneholder attributtene poststed og postnummer. Vi har lastet inn alle Norges postnr, og poststed ved hjelp av denne fila, http://portalservice.no/nor_postcodes.sql (gjorde noen modifikasjoner med fila) og de ligger nå inne i databasen. Denne inputfila inneholder alle postnr og poststed i hele Norge. Opprinnelig var postnr som string og da for å slippe å konvertere ble det også det i databasen vår. For å sjekke om et postnr er gyldig sjekker vi like godt opp mot hva listen med postnr. Finnes postnr der er det greit, hvis ikke så finnes det ikke. Egentlig hadde vi ikke tenkt til å ta med postnr og poststed, men pga at kartet trengte det for å finne koordinatene så var saken enkel. Grunnen til at vi har en egen entitet Post er fordi flere kunder kan ha samme postnr, og ved å gjøre det på denne måten slipper vi redundans.

Entiteten *frakt* inneholder attributtene fraktpris og grenseForFriFrakt. Her lagrer vi hva bedriften ønsker som fraktpris, og hva grensen for fri frakt. Når dette endres ved hjelp av GUI, oppdateres dette selvfølgelig, slik at også neste gang man starter programmet bruker man de nye verdiene.

Kodeforklaring:

Databasepakka:

Inneholder to filer. En fil for lesing og skriving til databasen (Databasecon.java), og en fil (Register.java), som inneholder metoder som brukes til søking i innholdet i Databasecon.java, samt litt andre nyttige metoder. Alle metodene i database er gjort statiske. Grunnen til dette er at vi kun ønsker å ha en aktiv forbindelse til databasen når programmet kjører. Dette er også grunnen til at constructoren er satt til private. En annen måte å løse dette på kunne vært og lagt opprettForbindelseMedDatabase() i mainklassen og kalt denne da programmet starter, men vi syntes det ble enklere og mer oversiktlig med klassenavn.metodenavn() enn objektnavn.metodenavn().

Alt av data som finnes i databasen blir lastet inn ved hjelp av Databasecon.java når programmet starter. Vi vurderte og kun laste inn synlige retter og bestillinger som verken er levert eller laget, for så å laste inn resten av dataene når vi først trengte det. Dette for å forhindre lang oppstartstid på programmet. Etter hvert kom vi fram til at vi uansett en eller annen gang i programmet hadde bruk for alt som lå i databasen, så for å beholde et raskt og effektivt program valgte vi å laste inn alt i starten. Ettersom programmet mest sannsynlig kun vil startes opp 1 gang pr. dag (når du kommer på jobb), så synes vi det er bedre å vente litt ekstra her, enn når kunden faktisk er på telefonen og ønsker å bestille. Ved at alt av data i databasen blir lastet inn i lister når programmet starter, blir jobber til findmetodene også mye enklere. For å forhindre at casting av exceptions spres seg har hver metode i Databasecon.java en try-catch blokk, slik at evt. Exceptions blir snappet opp her, og ikke sprer seg til resten av programmet.

Noen problemer som vi støtte på etter hvert har vært dette med sletting og endring av retter/kunder/bestillinger. Det bør jo være mulig å slette en rett, dersom man ikke lenger ønsker å ha den i menyen, men samtidig så kan det være gamle bestillinger som bruker denne retten. For å unngå dette problemet, har vi da lagt til et felt "synlighet", som sier om retten er på menyen eller ikke. Den blir altså aldri slettet fra programmet. Samme prinsippet med kunde, bortsett fra at en bestilling har en referanse til en kunde. Dersom denne kunden da blir slettet, har man et problem. Vi vurderte om vi i det hele tatt trengte muligheten for å slette kunde, og kom fram til at det vår greit å ha i forhold til personvern. Dette løste vi ved og først slette bestillingene (og ordrelinjene selvfølgelig) til denne kunden, for så å slette kunden. Konklusjonen er altså det og enten ta bort alt som har referanser til objektet, eller ingenting. Gjør du en mellomting blir det fort krøll. Et annet problem vi brukte mye tid på var det å endre rettene til en bestilling. Etter mye fram og tilbake ble løsningen som vi fant mest hensiktsmessig i forhold til databasen, å først slette de eksisterende rettene, for så å legge til nye.

For koblingen mellom databasen og javakoden har vi brukt JDBC. Her oppretter vi en forbindelse med databasen ved hjelp av en driver og bruker SQL setninger for å skrive/hente fra databasen. Vi bruker også et Statement som er den setningen som sendes til databaseserveren hver gang man leser/skriver fra databasen.

Valideringsmetodene i registerklassen er prøvd å gjort så generelle som mulig. I validerTekst()/validerTallOgBokstaverTillat() er det brukt regexp, ved hjelp av metode tekst1.matches([regexp]). Resten av valideringsmetodene sjekker bare om tall faktisk er tall, om postnr finnes i posttabellen, om leveringsdatoen er gyldig i forhold til hvilken mnd. vi er i osv.

ObjektTyper:

Inneholder en objektklasse for hver entitet i databasen, med unntak av Frakt. Hver klasse har da attributtene til entiteten som egenskaper til objektet. Alle objektene, unntatt Post, har en unik id for å skille de fra hverandre. Disse klassene inneholder også de metodene som trengs for å hente/sette data til et objekt. Relasjonen mellom kan du se ved hjelp av ER-diagrammet som ligger vedlagt. Entiteten Frakt valgte vi å ikke laste inn i systemet, men heller lese og skrive til databasen når dette trengtes. Dette fordi det brukes såpass sjeldent og når det faktisk skjer, så holder det at det blir rett. "Make the common case fast and the rare case correct" – Hennesy & Patterson. Alle objektklassene er relativt like, et unntak er klassen Bestilling. Her har vi laget leveringsdatoen til en bestilling som et GregorianCalendar objekt, som gjør sorteringen av bestillingene enklere.

testFiler:

For å teste programmet og metoder underveis har vi i tillegg til vanlig mainklasse, brukt junit tester. Disse er lagt i pakka testFiler. Denne er igjen delt opp i to klasser, en testklasse for hver av klassene i databasepakka. Målet med dette var for å se om det faktisk fungerte, og som gjør det enklere å tenke på spesialtilfeller. Stort sett alle add(), update() og find() metodene er testet. Det som er verdt å merke seg er at etter vi la inn test for å sikre oss at ikke samme tlfnr kan legges til to ganger, må man endre dette for hver gang man kjører testen.

Kart forklaring:

Vi hadde en del oppstartsproblemer med tanke på kartet. Ingen på gruppa hadde vært borte i noe lignende før og vi famlet litt i blinde til å begynne med. Først begynte vi å se på hvordan netbeans implementerte google maps for å finne koden som var nødvendig for å få opp et kart. Det viste seg at dette var ganske rotete og klønete, gikk etter hvert over til å bruke QT for å implementere kartet. QT viste egentlig bare en html side i et eget panel, og vi fikk da problemer med å markere leveringsadressen. Etter mye om og men gikk vi over til å bruke JXmapKit for å få frem kartet.

Vi bruker OpenStreetMaps (OSM) for å fremstille kartet gjennom et JXmapKit objekt. Vi har en egen metode som henter inn koordinatene (lengde og bredde gradene) til leveringsadressen (fra bestillingen). I denne metoden bruker vi Yahoo placefinder for å få en XML side hvor blant annet koordinatene står, og henter ut disse ved hjelp av en document parser. Så har vi en annen metode som tar inn koordinatene for å fremstille markøren som skal være på leveringsadressen.

GUI forklaring:

Vi valgte å bruke javas innebygde swing GUI komponenter i vårt program. Swing GUI Componentes inneholder både knapper, felter, tabeller og alt annet vi trengte i programmet vårt. Vi har bygd opp programmet slik at vi har et vindu der vi viser forskjellige panel, alt etter hvor i programmet brukeren befinner seg. I de forskjellige panelene ble swing komponentene plassert. Siden de forskjellige panelene blir bygd opp på samme måte valgte vi å ha en superklasse Panel.java som extender JPanel

med en konstruktør som setter layouten til panelet, kjører metodene `oppretteKomponenter()` og `leggTilKomponenter` og legger panelet til vinduet siden dette må gjøres for alle panelene. `Panel.java` inneholder de abstrakte metodene `oppretteKomponenter()`, `leggTilKomponenter()` og `nyttPanel()` siden man både må opprette og legge til komponenter og kunne gå videre i programmet til et nytt panel i alle panelene. I tillegg i hvert panel har vi metoder som er spesifikke for det enkelte panelet, men ikke alle panelene. I enkelte panel tar vi også inn informasjon fra panelet som oppretter panelet, som f.eks. `PanelKunde.java` eller `PanelBestillRett.java`.

For å plassere swing komponentene rett i forhold til hverandre valgte vi å bruke `GridBagLayout` som layout i panelene. `GridBagLayout` blir beskrevet som ”a sophisticated, flexible layout manager” (<http://download.oracle.com/javase/tutorial/uiswing/layout/visual.html>). `GridBagLayout` viste seg å ha både negative og positive sider. For det første var den ikke veldig lett å sette seg inn i til å begynne med. I begynnelsen ble det mye gjetting og prøving og feiling for å få komponentene plassert sånn som man ville ha dem. Men etter hvert fikk vi mer erfaring med bruk av denne layouten og da fikk vi stort sett plassert komponentene der vi ville med en gang. Vi fikk også utnyttet fleksibiliteten i layouten ved at vi stort sett kunne plassere komponenter akkurat hvor vi ville ha de i panelet, en mulighet vi sannsynligvis ikke ville hatt hvis vi hadde brukt en layout som kanskje ville vært lettere å sette seg inn i, men som ikke ville hatt like store muligheter for plassering av komponenter.

Vi valgte å lage en klasse `Constraints.java` som extender `GridBagConstraints` der vi har metoder for å sette posisjonen og andre constraints egenskaper. Ved å ha dette blir det lettere å lese hvilke constraints som er satt til et gitt tidspunkt. Ved å sette en og en egenskap ved en constraints blir det fort uoversiktlig og vanskelig å se ut fra koden hvor komponentene er plassert.

Vi valgte også å ha en klasse `Opprette.java` som oppretter forskjellige komponenter. Dette fordi man ofte gjør det samme når man oppretter en komponent. For eksempel har alle knappene en `ActionListener` og da blir det mer oversiktlig å ha en metode som setter `actionlistenere` i stedet for å måtte sette `actionlistenere` for alle knappene manuelt i `oppretteKomponenter`-metoden.

Vi prøvde oss på å bruke MVC- Model- Viewer – Controller, men kom fram til at tidsrammen på prosjektet ikke tillot oss dette. Det ble for komplisert og ettersom programmet er så lite, ble andre ting prioritert. Fordelen med og hatt en MVC er jo det at det er enklere for en utenforstående person å komme inn å skrive på GUI uten å måtte ta hensyn til kode som ikke har noe med GUI å gjøre. Man får skilt de forskjellige ”delene” av koden litt fra hverandre, noe som vi ville prioritert mer, dersom programmet skulle realiseres. Panelene kaller derfor metoder fra `Databasecon` og `Register` og sender informasjon direkte til databasen.

Et problem som egentlig har fulgt oss gjennom hele prosjektet har vært det med oppløsning. Første møte med dette ble under midtveispresentasjonen, da prosjektoren ikke hadde samme oppløsning som skjermen vår. For å forhindre dette problemet har vi nå gjort slik at komponentene skalerer, avhengig av plassen på skjermen, samt satt en `minimumsize` på vinduet. Denne sizen er så liten at vanlige 15” maskiner med WXGA oppløsning ikke har noen problemer med å takle dette. Vi har sagt at et minimumskrav til skjermene som programmet skal kjøres på er WXGA ettersom vi vurderer dette som mest hensiktsmessig.

Vi valgte som tidligere sagt å ha passord på knappene ”Fraktpris/Passord” og ”Administrer retter”. Vi valgte å ha et `rootPassord` som kun systemansvarlig for programmet kjenner til, og et vanlig passord

for de ansatte. Det er ikke foretatt noen kryptering på passordet i databasen. Dette er noe vi ville prioritert dersom programmet faktisk skulle realiseres.

JARS

For å få tilgang til databasen ved hjelp av JDBC, trenger vi en JAR fil som hjelper oss med dette. For å vise kartet bruker vi JXMapKit. For å få tilgang her trenger vi to JAR filer. Jar filene vi har lagt til i prosjektet er Mysql-connector-java-5.1.13-bin, swingx-1.6 og swingx-ws-2010_10_10.

Parprogrammering

Før prosjektet startet ble vi sterkt oppfordret av en studentassistent til å prøve parprogrammering. Vi bestemte oss for å prøve dette mer aktivt etter evalueringen av 1. sprint, der det kom fram at det hadde vært for mye oppdeling av oppgaver og lite samarbeid innad i gruppa. To gruppemedlemmer satte seg da ned foran en pc, og byttet på å skrive kode/si hva som skulle skrives. Det var spesielt i forbindelse med GUI at parprogrammering ble gjennomført. Grunnen til dette var at swing er en litt annen måte å programmere på i forhold til vanlig java, og det var derfor her det var størst behov for samarbeid for å skjønne koden. Spesielt i oppstartsfasen med swing var parprogrammering veldig nyttig. Alle fikk kjennskap til oppbygning, syntaks, og metoder til de aktuelle funksjonene.

Parprogrammering ble mye brukt i vårt arbeid. Ikke bare gir det innsikt i hverandres kode, men du lærer også å gjøre ting på andre måter enn du vanligvis ville gjort selv. Det er viktig å kunne se en sak fra flere sider, samtidig som det er en fin måte å gi andre i gruppa innsikt i din kode. Et lite startproblem som gjorde det vanskelig å parprogrammere var dårlige variabelnavn, slik at kun den som hadde skrevet koden skjønte noe. Javadoc var det også dårlig med i starten, så heller ikke den var mye til hjelp. Etter evalueringen av 1. sprint ble derimot dette tatt opp, koden ble forbedret og det ble enklere for alle i gruppa å gjøre endringer i koden. Vi har også prøvd trippelprogrammering, rett og slett for å få litt mer input, dersom det var spesielle deler av programmet som skulle avgjøres. F.eks. i forhold til brukergrensesnitt, plassering av elementer, tekstinhold osv. Dette gav oss flere meninger, som gjorde at vi ble mer bevisst på hvordan programmet skulle se ut i forhold til vår målgruppe.

En litt annen måte å parprogrammere gjorde vi egentlig uten baktanke på tirsdagsmøtene. Tirsdagsmøtene har pleid å vært to timer hvor alle møtes og jobber samtidig. Dette for å kunne diskutere og samarbeide om aktuelle problemer man evt. skulle komme opp i. På disse møtene var det ofte ting som ikke var helt ferdig testet, men som allikevel var commitet til den endelige versjonen på SVN, og som da folk kunne si i fra at alt ikke funket som det skulle. Dette problemet ble da fikset i plenum, som igjen gav resten av gruppa innsyn i kode.

Oppdatering mellom maskiner

Ettersom de ansatte, kokk og leverings bud mest sannsynlig vil sitte på forskjellige datamaskiner må det være en form for synkronisering mellom pc'ene. Dette har vi ordnet ved at en timer kaller en

oppdateringsmetode hvert 10. sek og oppdaterer bestillingene hos kokk og leveringsbud. Listene blir så sortert på nytt, før tabellen tømmes og fylles ut igjen med bestillinger. Det er kun på bestillinger at det er viktig med kontinuerlig oppdatering. På rettene i menyen blir det kun synkronisert når programmet starter opp.

Brukertesting

For å teste brukervenligheten til et program bruker man ofte brukertesting. Det som er viktig er å teste programmet på en aktuell bruker fra målgruppen. Dette er jo eneste måte å se hvordan personene som skal bruke programmet opplever det. Måten vi har utført brukertesting har vært delt opp i fire deler som hver testperson skal utføre:

- 1) Lag en ny bestilling
- 2) "Vær en kokk" og si at du har ferdiglagd en bestilling.
- 3) "Vær et leveringsbud" og lat som om du skal kjøre ut en bestilling
- 4) Test mulighetene i admin menyen.

Vi som har lagd programmet har ikke lov til å svare på spørsmål under selve testen. Det er også viktig at testpersonen snakket høyt når han/hun utfører oppgavene. På denne måten får vi innblikk i hva en utenforstående synes om løsningen vi har lagd. Etter testen kan vi svare på spørsmål fra brukeren, og ta generelle kommentarer om programmet. Brukertesting har hjulpet oss mye, spesielt i forhold til plassering av knapper, tekst o.l. Vedlagt finner du resultatene fra tre brukertestinger utført på tre forskjellige personer: En informatikkstudent, en fra energi og miljø, og en fra biologi. Alle tre brukertestene er utført på forskjellige stadier i prosessen.

Utvidelser

Dersom programmet skulle realiseres er det selvfølgelig ting vi ville utvidet/endret. Vi ville endret slik at flere funksjoner kunne utnyttes bedre ved hjelp av tastaturet, ved f.eks. at når man trykker "ENTER" så trykker man automatisk på knappen på dette panelet. Annen ting er å implementere MVC som ville gjort fremtidig vedlikehold og utvidelse av programmet enklere. Da kan utenforstående komme inn og slippe å forstå hele koden for å gjøre en endring. En annen ting er å legge til et felt antall i OrdreLinje, slik at man slipper styret vi har gjort i panelBestillRetter og panelBestillLevering. En annen viktig ting er kryptering av passord i databasen.

Avslutning

Som du nå har sett har vi i dette prosjektet laget et program for en pizzasjappe. Læringskurven har vært bratt, og fra å være helt blank på enkelte sentrale områder, som f.eks. Swing, har vi nå opparbeidet oss et grunnlag og en forståelse for videre arbeid med dette. Ettersom grunnlaget før prosjektet, spesielt på JDBC og Swing, var så som så, har det vært mye prøving og feiling. En gjentakende fremgangsmåte har vært å gjøre ting på en måte for så å endre det uken etter, når vi

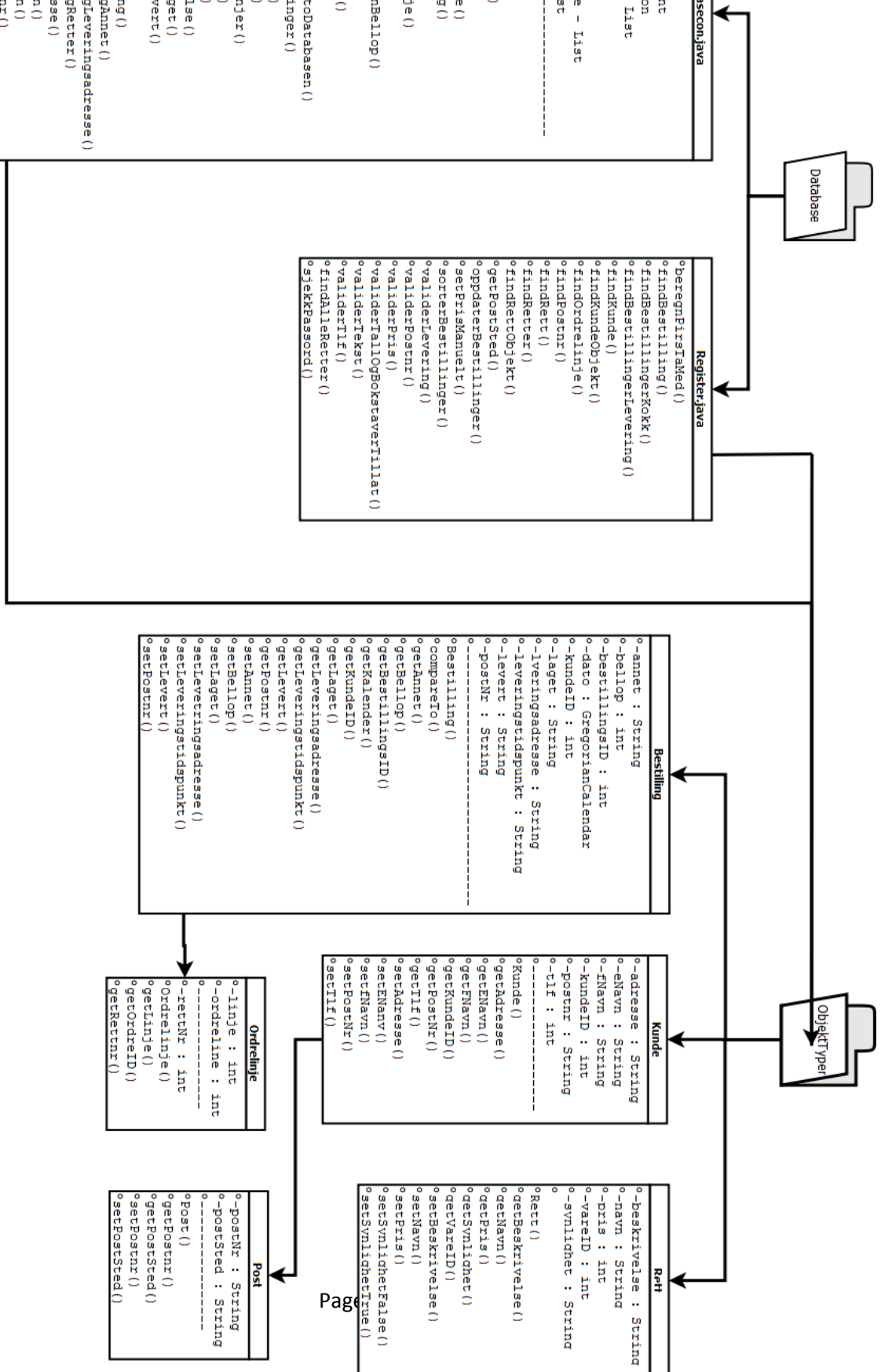
fant ut at det kunne gjøres enklere. Men, prøving og feiling er jo en av de metodene som man lærer best ved, selv om det er tidkrevende og til tider frustrerende.

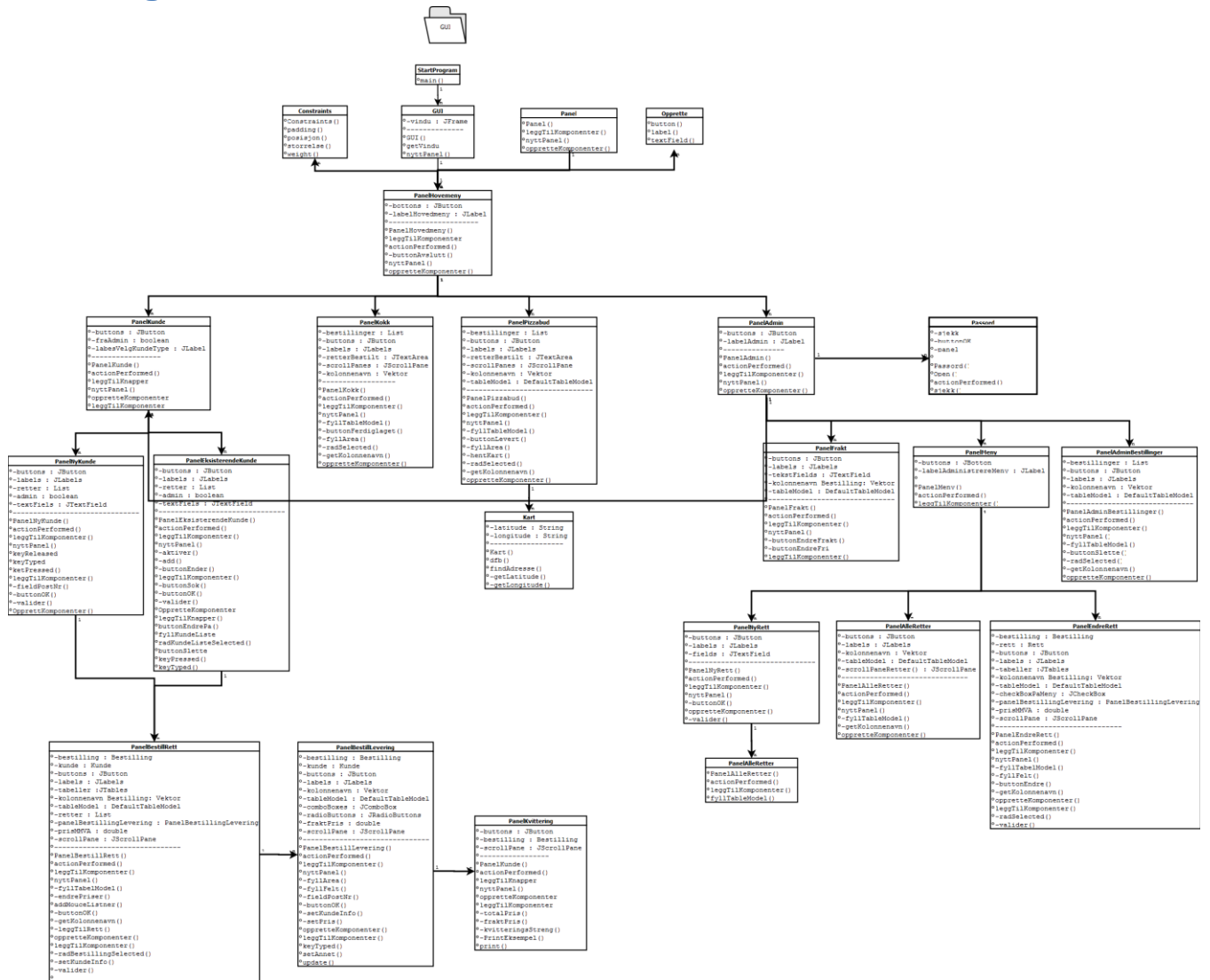
Selve scrum-arbeidsmetoden har fungert delvis. Det med to ukers sprinter, prioritering av funksjonalitet, og jevnlige møter har fungert veldig bra, men selve utfordringen slik vi ser det, handler om å beregne riktig tidsbruk på oppgaven. Det har alltid vært en sprint hvor ting har vist seg at ting tar lenger tid enn planlagt, og at ting da blir forskyvet. Heldigvis er det andre ting som tar kortere tid enn beregnet, men det går likevel i negativ favør. Ettersom det var første gang vi brukte scrum er dette sikkert også en tilvenningssak, slik at vi forhåpentligvis blir bedre til å beregne tid neste gang vi bruker scrum.

Selv er vi veldig fornøyd med programmet. Vi mener selv vi har oppfylt kravspesifikasjonen og er fornøyd med det programmet vi leverer, på tross av at vi bare er tre på gruppa. Selvfølgelig er det vanskelig å sette strek, for så å finpusse på det man allerede har laget, men vi mener at vi har funnet en fin balansegang.

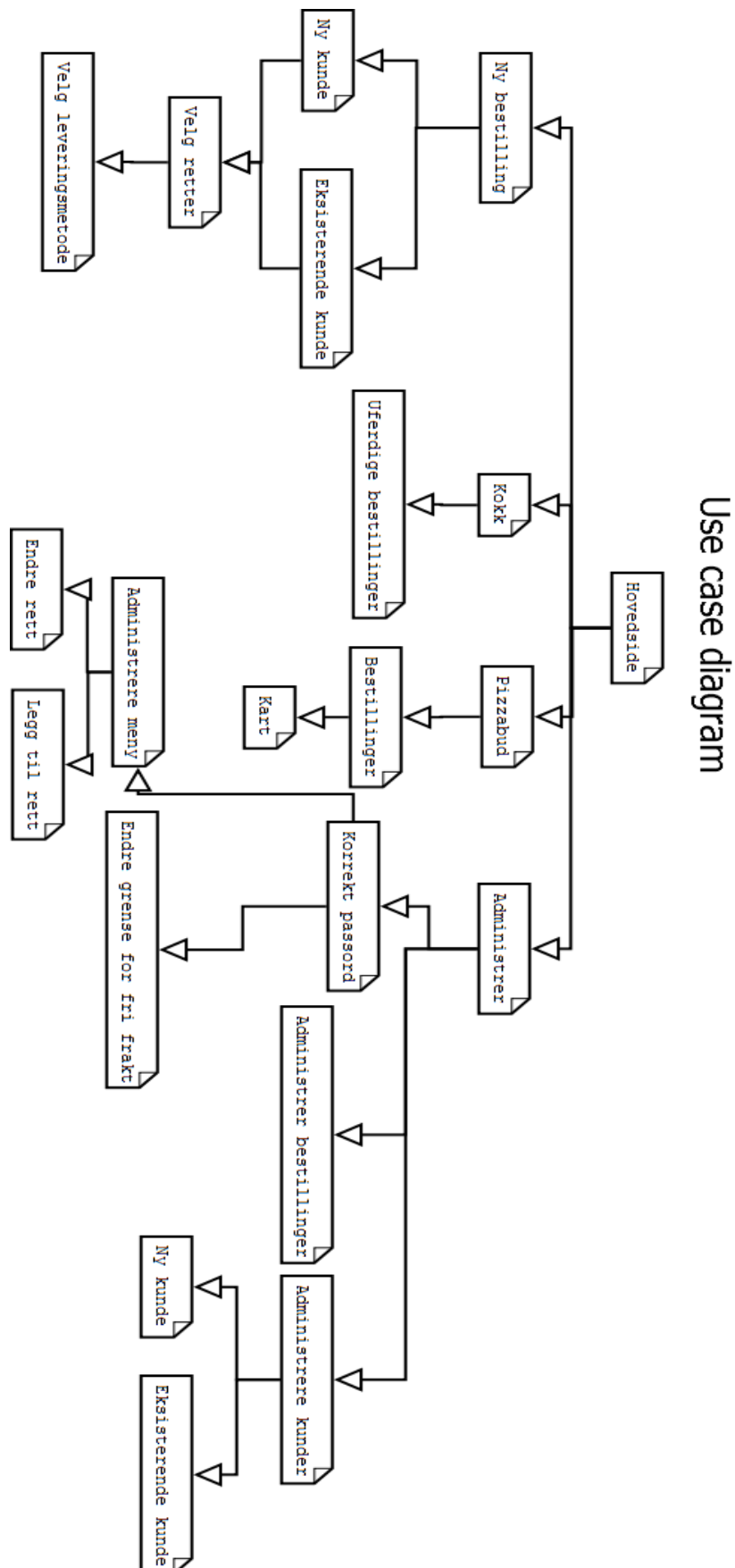
Appendiks A:

Alle vedleggene ligger også i mappen "Vedlegg" for større versjon

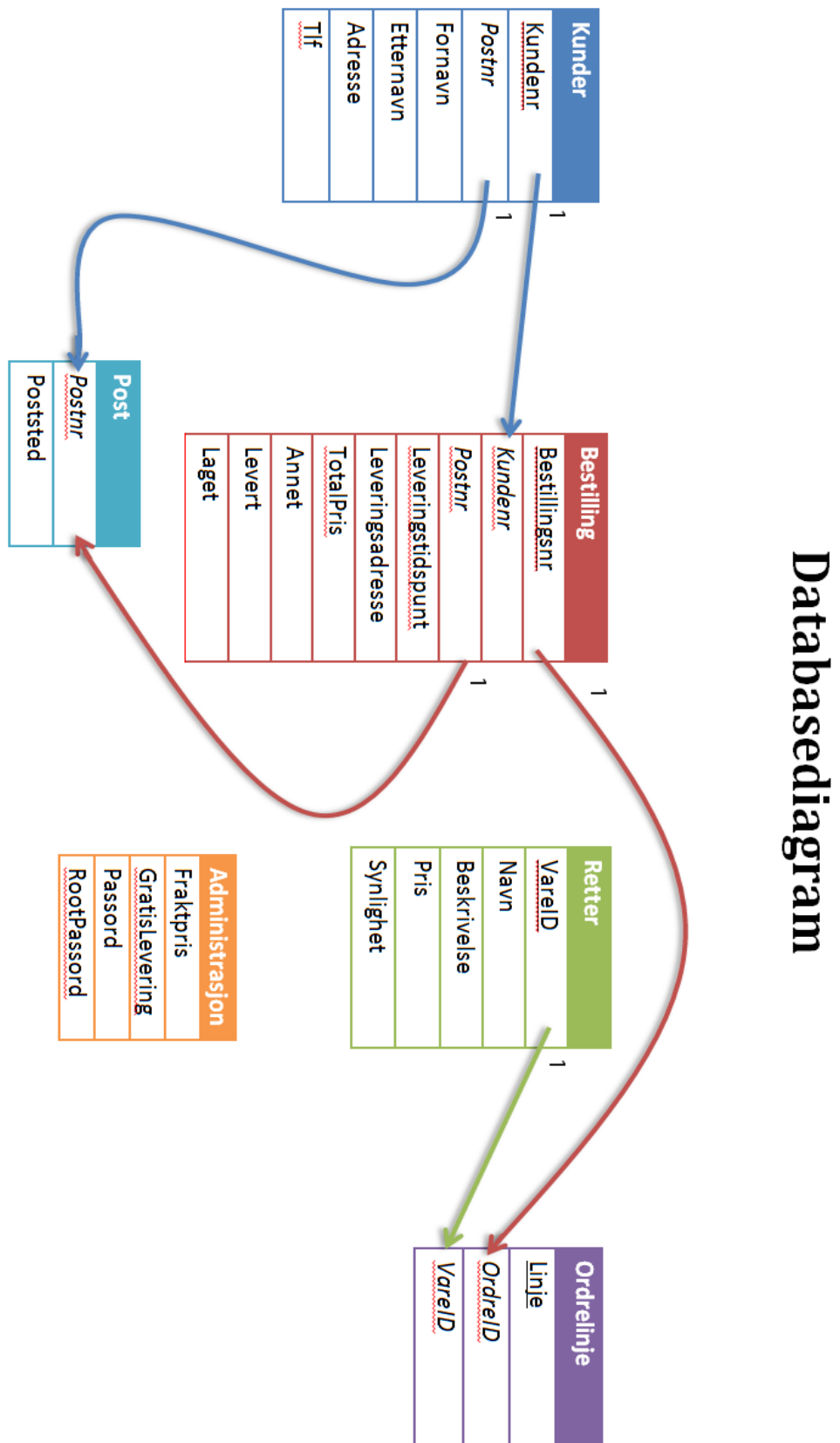
[illegible]



Use case diagram



ER- diagram



Appendiks B:

Møtereferater

Møtereferat 07.09.10

Tilstede: Haakon, Torunn og Sindre

Fraværende: Christoffer

Hva har vi gjort:

- Avtalt hva som skal gjøres
- Forberedt spørsmål til studass
- Ordnet dropbox til prosjektet
- Sendt mail til studass ang. nytt møte
- Teambuilding
- Laget utkast til usecase-diagram og ER-diagram

Møtereferat 10.09.10

Tilstede: Torunn, Haakon

Hva har vi gjort:

- Koblet eclipse SVN og googlecode slik at synkronisering fungerer og alle har tilgang til koden
- Laget 1. utkast til Scrum planen (se vedlegg s 27)
- Begynt å tegne GUI (se vedleggsmappa)
- Laget første utkast på backlog
- Skissert arkitektur
- Fordelt oppgaver.
 - GUI – Hovedfokus Torunn
 - Sette seg inn i GUI
 - Lage et utkast til å legge til ny kunde
 - Database – Hovedfokus Haakon
 - Finpusse på arkitekturtegningen
 - Opprette databasen
 - Opprette forbindelse med programmet og databasen
 - Begynne å lage metoder for å hente/skrive til databasen hvis tid
 - Kart- Hovedfokus Sindre
 - Se på hvilke muligheter som finnes
 - Se litt på hvordan dette kan inkluderes i GUI'et
- Satt deadline,

Møtereferat 24.09.10

Tilstede: Haakon, Sindre og Torunn

1. iterasjon er ferdig. Hva har vi gjort:

- Møte med studass
 - o Diskutert framgangen til nå
 - Kommet litt kort
 - Må rydde i klassene i databasen. Kan ikke ha tre connections til samme databasen som tidligere tenkt
 - Kart er ikke ferdig
 - Små barnesykdommer med GUI'et, eks. popups.
 - o Diskutert planlagt framgang videre
 - rydde i kunde, rett, bestilling
 - rydde i connectionklassen
 - GUI få inn element (tekstfelt, knapper...)
 - GUI ting åpner seg i samme vindu
 - o Fått kommentarer til arbeidet så langt. Tips til forbedringer
- Skissert GUI og programflyt
 - o DB – Java – Listner – GUI, fokus på listner i neste iterasjon

Evaluerings av 1. iterasjon:

- o For slapt!
- o Litt for liten innsats
- o Skal ta i ett tak til neste fredag
- o Kommet litt for kort i forhold til andre grupper
- o Litt for dårlig organisering, litt for mye "solo run". Prøve med parprogrammering og litt mer gruppearbeid, og håpe på den måten at motivasjonen for å arbeide øker.
- o Huske å skrive javadoc med en gang. Tungvint for andre å lese koden uten.

Møtereferat 01.10.10

Tilstede: Torunn og Haakon

Hva har vi gjort:

- Jobbet videre med tingene vi ble enige forrige fredag.
- Møte med studass:
 - o Sett over database, gitt tips til arkitektur.
 - o Sett over GUI, godkjent framgangen til nå.

Til neste gang (8. okt 12:00):

- Huske å skrive javadoc NÅR vi skriver metodene, ikke etterpå!
- Rydde i kode, noe som er gjort litt tungvint
- Tilpasse arkitekturen
- Lite resultater å vise til med kartet
- kobling mellom GUI og objekter

- Klar for demo og fremvisning for de andre gruppene

Møtereferat 08.10.10

Tilstede: Haakon, Sindre og Torunn

2. iterasjon ferdig.

Hatt møte med studass, fikk beskjed om:

- Rydde opp, databasen skal være static, flytte metoder som ikke skal være i databasen. Ha en egen registerklasse.
- GUI: Fornuftige variabelnavn.
- Egentlig tenkt MVC, men kom fram til at prosjektet er for lite og at MVC er for komplisert at det ikke var verdt å bruke den dårlig, men fungerende løsningen vi hadde.
- Dette vil føre til at GUI og databasen ikke er så adskilt som ønskelig, men
- Kart, har nå fått opp kart ved hjelp av QT. Får vise det vi har på framføringen. Legge inn adresse og markør senere.

Evaluerer etter 2. iterasjon:

- Fortsatt bak, men bedre. Gjør veldig mye på en tungvinn måte først, for så å gjøre det om igjen etterpå. Tar mye tid...
- Innsatsen har steget litt, men ikke nok til at dette skal bli det resultatet vi håper på.
- Parprogrammering er ålreit! Får innsikt i hverandres kode, samt innføring i hvordan det andre personen tenker.
- I forhold til backlog ligger vi bak. Ting tar lenger tid enn hva vi beregner og ting "flyter" over til neste sprint.

Møtereferat 22.10.10

Tilstede: Sindre, Haakon og studass

3. iterasjon ferdig.

Hva har vi gjort siden forrige møte:

- Fått opp kart ferdig finpusset
- Alle paneler laget
- Utvidet databasen og register klassene med nesten alle de metodene som kreves for å oppfylle kravspesifikasjonen
- Parprogrammert i forbindelse med GUI
- Trippelprogrammert – litt mer ineffektivt enn parprogrammering, men alle lærer allikevel mye.

Hva skal vi gjøre til neste gang:

- Gjøre GUI-kode mer effektiv, unngå redundans
- Begynne så smått å skrive rapport
- Kommentere skjermbildene og gangen i systemet, finpusse ER diagram, finpusse use-case diagram og begynne på klassediagram

Evaluerer etter 3. iterasjon:

- God arbeidsperiode
- Motiverende å se hvordan de andre har laget det
- Bra innsats!
- Kommet over "kneika", som gjør at det er enklere å jobbe, både mer og mer effektivt.
- Føler vi er kommet greit i forhold til de andre gruppene.
- Mye finpuss igjen. Har fortsatt en god del å gjøre før et ferdig produkt
- Kommet litt for kort på rapporten foreløpig.

Møtereferat 05.11.10

Tilstede: Haakon, Sindre og Torunn

4. iterasjon ferdig.

Kommentarer etter 4. iterasjon:

- Det meste er ferdig
- Bra arbeidsperiode, nesten på høyde med iterasjon 3!
- Finpussing på GUI
- Mye rapport igjen, kommentere vår egen kode
- Føler vi kommer i mål
- Skal teste programmet på fremmedperson i løpet av neset uke (eget referat, se vedlegg s xx)

Til tirsdag 09.nov:

- GUI skal være ferdig
- Exe ting og ikon
- Finpuss på diagram, backlog og oppdateringsmetoder

Til neste gang:

- Finpuss på GUI
- Rapport
- Oppdatere diagrammene
- Finpusse kode
- Ferdig!

Foreløbig plan via Scrum:

Møtetidspunkt for gruppa:

- **Tir 10-12**
- **Fredag kl 12, hvis det er forelesning. Ellers kl 10**
- **Vi har også studassmøte på fredagene/annenhver kl 12. Evaluering av hver iterasjon da.**
- **2 uker iterasjon**

1. iterasjon:

- Sette opp databsen - Haakon
- Begynne å sette seg inn i gui –Torunn
- Kart – Sindre
- Kode controller på fredag. Mellom GUI og java
- Punkt 1 først i rapportbeskrivelsen først.
- Prosjektrapport, huske å skrive notater

Deadline: 24. sept!

2. iterasjon:

- Koble sammen, gui, databser
- Begynne med parprogrammering
- Skrive koden mellom Guiet og databsen
- Punkt 2 i prosjektbeskrivelsen
- Prosjektrapport!

Deadline: 7. okt!!

3. iterasjon:

- Punkt 3 i rapporten
- Koble kart
- Prosjektrapport

Deadline: 22. okt!!

4. iterasjon:

- Frakt – oppfylle kravspekk
- Rapport!
- Legger inn slingringsmonn her, ettersom ting tar ofte lenger tid enn man tror, eller det kommer uforutsette problemer.

Deadline: 5.nov!!

5. iterasjon:

- Finpuss av kode
- Rapportskriving

Deadline: 18. nov!!

Kommentarer etter midtveispresentasjon

Lå greit an i forhold til de andre gruppene. Fikk følgende tips:

- Mer utfyllende kvittering med bla. Momsberegning og stash
- Admingrensesnitt som kan legge til/fjerne retter, slette kunder, annullere bestillinger, justere pris manuelt i forb. Med rabatt
- Endring av hovedmeny, navn på knapper, tydeligere fordeling mellom kjøkken, kasse, levering og admin.
- Feilmelding hvis noe går galt. Popups, gjøre dette på tirsdag.
- Hva skjer hvis programmet skal kjøre på flere pc'er? – **Spørre studass på fredag hva kravet er**
- Oppdatering av antall varer. Hvordan?
- Mulig å bytte plass på pris og beskrivelse i leggTilRettPanel og gjøre beskrivsetekstboksen litt større?
- Oppløsning!!! Viktig

Brukeresting:

Daniel 03.11.2010

“Dette er konkret sitering fra testpersonen”

- “Tungvinn bestillingsprosess”, ork å legge inn bestilling, ved å få nåværende mnd og dag færdig utfylt vil mye være gjort..
- Oppdaget at man kan bestille bestillinger fra tidligere datoer i år. Dette må fikses!
- Litt uoversiktlig på velg leveringstidspunktpanelet, mye elementer. Tar lang tid å sette seg inn i.
- Fikse samme ting på postnr i leveringsadresse i bestillingsprosessen, som i nykunde ?
- “Hvor står prisen” på velgrettpanelet i bestillingsprosessen? Trenger vi denne?
- Vil gjerne ha prisen på hver rett på kvitteringa og sum rett under dette. Ellers pen kvittering
- “Vanskelig å skjønne at endreknappen” fungerer, både på endre kunde og rett. Dette må endres.
- “Irriterende at alt du har skrevet på bestillingen (leveringstidspunkt, annet, leveringsadresse osv) ikke blir lagret når du ønsker å endre rett”.
- Leveringsadresse kan ikke inneholde tall – Endre validerLeveringsadresse()
- Rart at vi kun kan søke etter kunde på tlf nr. Hva hvis man ikke kan morens tlf nr? Bør ha navn også.
- Stusset på admin kunder, at man da gikk videre til bestilling, men dette skal vi jo fikse med et nytt panel.

Som konklusjon sa han at programmet var veldig bra og at tingene han har sagt her er småpirk.

Kari 11.11.10

Hovedmeny: Skriften på knappene kunne vært større

PanelKokk:

- Skal kokken kunne trykke på retten og se hva den inneholder?
- Når ferdiglaget; tøm rettene-feltet.
- Når en rett er ferdiglaget sett valgte rad som den øverste raden automatisk?
- Skal den hentes eller leveres? Må lages ferdig tidligere hvis den skal leveres til et tidspunkt.

PanelPizzabud:

- Hva er egentlig annetfeltet i bestilling?
- Ha med med kundenavn i tabell.
- Trenger ikke bestillingene som hentes her. Bare de som skal leveres.
- Mindre info i tabellen. Ha felt med info i.

PanelBestillRett:

- Mindre bestillingstabell
- Priser under bestillingstabell

PanelBestillingLevering:

- Litt uklart hva kundens adresse/leveringadresse er. Innrykk på disse eller større skrift på hentes/leveres.
- Priser sammen med oversikt over retter.
- Standardstørrelse på bestilte retter-feltet.
- For eksempel hvis man ønsker ekstra ost på rett1 må man vente til PanelBestillingLevering med å si dette? Burde det være et annet felt i BestillRett også? (eventuelt for hver rett)? Et annet-felt med info for kokken og et annet-felt med info for leveringbud?

PanelFjernRett:

- Fjern knapp i stedet for OK

Tilbakeknapp i eksisterende kunde.

Emil 15.11.2010

- Overskrift på PanelBestillingLevering, noe som Leveringsdetaljer eller noe sånt.
- Vanskelig å skjønne at leveringstidspunkt klokkeslettet skal settes der det skal settes, trodde han satte klokkeslettet ved datoen. Sette klokkeslettet til automatisk åpningstiden eller systemdatoen.
- Klønete bare å kunne søke opp kunder via telefonnummer både på admin siden og bestilling siden.

Ellers ser programmet veldig bra ut, og han mente at programmet kunne selges til en nyoppstartet bedrift som kunne være interessert i en billig løsning framfor en mye mer fancy løsning til mye mer penger.

Appendiks C:

Risikoanalyse

<i>Risiko</i>	<i>Sansynlighet</i>	<i>Konsekvens</i>	<i>Viktighet</i>	<i>Tiltak</i>
Datakrasj	1	3	3	Backup
Sykdom	1	2	2	Dele informasjon
Torunn ikke tar faget	1	3	3	Innsats
Manglende innsats	1	3	3	Innsats
Manglende oppmøte	1	2	2	Planlegging
Tap av kode	2	3	6	Backup

Timeliste:

Torunn Sæther:

Dato:	Fra:	Til:	Ant Timer:	Ka:
07.09.2010	10	12	2	Møte
10.09.2010	12	2	2	Møte
14.09.2010	10	12	2	Begynne med gui
17.09.2010	10	12	2	Møte
21.09.2010	10	12	2	Kunne hente data fra guiet
22.09.2010	9	12	3	kundepanel
24.09.2010	10	12	2	Møte
27.09.2010	10	15	5	rettpanel
28.09.2010	10	12	2	Møte/jobbing
28.09.2010	14	17	3	rett/kundepanel
30.09.2010	10	13	3	bestillingspanel
01.10.2010	10	13	3	Møte/GUI
02.10.2010	12	14	2	gui
04.10.2010	10	14	4	gui: endringer av meny
05.10.2010	15	17	2	gui: feilretting finpuss på det som er gjort
07.10.2010	10	12	2	gui
08.10.2010	11	14	3	Jobbing + møte
09.10.2010	12	14	2	gui diverse
11.10.2010	12	15	3	gui diverse
12.10.2010	10	12	2	gui diverse
13.10.2010	10	12	2	Finpussing før presentasjon
15.10.2010	12	14	2	evaluering etter midtveispresentasjon
18.10.2010	11	15	4	PanelKokk og PanelPizzabud
19.10.2010	10	12	2	Parprogrammering
20.10.2010	10	13	3	Parprogrammering
20.10.2010	13	14	1	Rydde i kode
25.10.2010	12	15	3	fraktadmin
26.10.2010	10	12	2	Rydde i kode

27.10.2010	9	13	4	Tilpasse dårligere oppløsning
29.10.2010	9	10	1	Rydde i kode
29.10.2010	10	13	3	Parprogrammering
30.10.2010	10	17	7	Adminbestillinger, mulighet for å endre bestilling + rydding i kode
02.11.2010	10	12	2	timer i panelkikk og panelpizzabud
02.11.2010	14	16	2	gui: finne best mulig plassering av komponenter i panelene
03.11.2010	10	13	3	gui: størrelse på knapper , skrift osv
04.11.2010	10	12	2	gui: størrelse på knapper , skrift osv
05.11.2010	10	14	4	Møte, rapportskrivning, finpussing
06.11.2010	12	14	2	Dele opp bestilling i to paneler
07.11.2010	12	15	3	Dele opp bestilling i to paneler
08.11.2010	13	15	2	finpussing
08.11.2010	22	24	2	finpussing
09.11.2010	10	12	2	javadoc
09.11.2010	21	24	3	javadoc og kommentarer
10.11.2010	10	13	3	finpussing
10.11.2010	20	24	4	Feilsøking og feilretting
11.11.2010	9	14	5	Finpussing og feilretting
11.11.2010	18	20	1	Brukertesting + finpussing
12.11.2010	11	14	3	Finpussing + møte
13.11.2010	17	19	2	finpussing
14.11.2010	13	17	4	Finpussing og feilretting
14.11.2010	19	22	3	finpussing
15.11.2010	11	14	3	Finpussing + rapportskrivning
15.11.2010	22	24	2	Lage søkemulighet for kunde ved navn
16.11.2010	10	12	2	Finpuss
16.11.2010	14	17	3	Utskrift + passord
16.11.2010	19	21	2	Finpuss
17.11.2010	10	14	4	Finpuss
18.11.2010	12	14	2	Finpuss

Haakon Sønsteby:

Dato:	Fra:	Til:	Ant Timer:	Ka:
7. sep.	10	12	2	Møte
10. sep.	12	14	2	Møte
12. sep.	17	18,30	1,5	Begynte med databasen, plundring..
14. sep.	10	13	3	Fikk til databasen
19. sep.	12	14	2	Justert databasen
22.sep.10	12	14	2	Justert databasen
23.sep	18	19,30	1,5	Fiklet på databasen
24.sep	12	14	2	Møte
28.sep	12	14	2	Rydde i databsen
29.sep	19	22	3	Objekter, søkealgoritmer

30.sep	12	17	5	Objektorientert
01.okt	10	13	4	Møte/databasejobbing
01.okt	18.30	21	2,5	Rydding i databasen
04.okt	14	16	2	Databse
05.okt	10	12	2	Møte
06.okt	10	12	2	Lyttergrensesnitt
08.okt	12	14	2	Møte
09.okt	10	12, 30	2,5	Databse, rydding, statisk
10.okt	16	18	2	Skrevet tester
14.okt	19	21	2	Justert databasen
15.okt	12	14	2	Jobbet med databasen + evaluering etter midtveispresentasjon
19.okt	10	12	2	Parprogrammering
20.okt	10	13	3	Parprogrammering
22.okt	10	13	3	JavaDoc
24.okt	19	21	2	Posttabell og metoder
25.okt	18	20	2	Valideringsmetoder
26.okt	10	14	4	Bugfixing
28.okt	17	18	1	Leveringsmetode og kart
29.okt	10	13	3	Parprogrammering
30.okt	15	17	2	Sorteringsmetode
31.okt	19	21	2	Testing og fikse oppdateringer på flere maskiner.
01.nov	17	19	2	Fortsette med oppdatering på flere maskiner. Bruk av timer, ikke knapp
02.nov	10	13	3	Finpusning
04.nov	17	20,5	3,5	Jacadoc, backlog, testing og finpusning av kode
05.nov	10	14	4	Møte, rapportskrivning og finpusning
07.nov	13	16	3	Backlog oppdatering
08.nov	10	12	2	Klassediagram, finpusning av ER diagram
09.nov	10	14	4	Møte, brukertesting
10.nov	13	17	4	Finpusning
11.nov	13	15	2	Feilsøking
12.nov	12	16	4	Møte, rapportskrivning og finpusning
13.nov	9	12	3	Debugging
14.nov	13	18	5	Rapportskrivning, klassediagram
15.nov	10	12	2	Testing av program
16.nov	10	5	5	Innsput av prosjekt
17.nov	10	15	5	Innsput av prosjekt
18.nov	12	14	2	Finpuss

Sindre Svendsrud:

Dato:	Fra:	Til:	Ant Timer:	Ka:
7. sep.	10	12	2	Møte
12. sep.	17	18.30	1,5	Tegne arkitektur
14. sep.	13	16	3	Tegne arkitektur
19. sep.	11	14	3	Tegne arkitektur/begynt på guiskisser
22.sep.10	12	14	2	Guiskisser/ Begynt på kart

23.sep	17	18	1	Guiskisser ferdig/forts. Kart
25.sep	12	14	2	Møte
27.sep	11	14	3	Kart
29.sep	19	22	3	Begynt på kart med QT
30.sep	12	17	5	Forts. Kart QT
01.okt	10	13	4	Møte
01.okt	15	17	2	Forts. Kart QT
04.okt	15	16	1	Forts. Kart QT
05.okt	10	12	2	Møte
06.okt	10	12	2	Lyttergrensesnitt
08.okt	12	14	2	Møte
10.okt	16	18	2	Lyttergrensesnitt
14.okt	17	19	2	Parprogrammering GUI
15.okt	12	14	2	Evaluering av midtveispresentasjon
18.okt	11	14	3	Begynt på kart med JXmapViewer
19.jan	10	17	7	Kart JXmapViewer og koordinater
21.jan	10	12	2	JavaDoc
24.okt	19	20	1	Kart nesten ferdig
25.okt	18	20	2	Valideringsmetoder parprogrammering
28.okt	17	18	1	Leveringsmetode og kart ferdig (parprogrammering på slutten)
29.okt	10	13	3	Parprogrammering GUI
30.okt	15	17,0	2	Sorteringsmetode
31.okt	19	21	2	Testing og fikse oppdateringer på flere maskiner. Parprogrammering
01.nov	17	19	2	Fortsette med oppdatering på flere maskiner. Bruk av timer, ikke knapp. Parprogrammering
04.nov	20	24,0	4	testing og finpusning av kode
05.nov	10	14	4	Møte, rapportskriving og finpusning
07.nov	13	14	1	Oppdatering av Use Case diagram
09.nov	10	14	4	Møte, brukertesting
10.nov	13	18,5	3,5	Finpusning
11.nov	13	4	1	Brukertest på Emil
12.nov	12	16	4	Møte, rapportskriving og finpusning
13.nov	9	12	3	Debugging
14.nov	13	7	4	Rapportskriving generelt + kart
15.nov	10	12	2	Testing av program
16.nov	10	5	5	Innsjutt av prosjekt
17.nov	12	15	3	Innsjutt av prosjekt
18.nov	12	14	2	Finpus

Backlog

<i>Pri:</i>	<i>Beskrivelse:</i>	<i>Sprint # 1</i>	<i>Sprint # 2</i>	<i>Sprint # 3</i>	<i>Sprint # 4</i>	<i>Sprint # 5</i>
1	Tegne arkitektur	x	x			
1	Opprette database	x				

2	Tegne skjermbilder	x				
2	Implementere GUI kunde	x				
2	Forbindelse mellom database og programmet	x				
3	Implementere kart	x	x	x		
3	Opprette MVC mellom database og GUI		x	x		
3	Forbindelse mellom programmet og GUI		x	x		
3	Implementere GUI meny		x			
4	Implementere oversikt over ikke effektuerte bestillinger			x		
4	Implementere oversikt over bestillinger som er leveringsklare			x		
4	Implementere muligheten for å legge til bestilling			x	x	
5	Administrere fraktpriser				x	
5	Administrere bestillinger				x	
2	Finpuss av program					x

Sprint1:

Pri:	Beskrivelse:	Demo:	Notes:
1	Tegne arkitektur	Tegne hvordan vi ser for oss gangen i systemet	Usikker om vi trenger postTabellen. Vi dropper den foreløpig, også heller legger den til etterpå
1	Opprette database	Skrive noe til den gjennom ssh for så å se om det blir lagret.	En sql database ble laget ved hjelp av mindTerm på login.stud.ntnu.no. Dette gikk greit da dette hadde vi gjort i et tidligere fag.
1	Tegne skjermbilder	Tegne hvordan vi ser for oss layout på skjermbildene	Noen ulike synspunkter og meninger. Vi valgte å fokusere på å få på plass alt først, så får vi brukerteste senere

Implementere GUI kunde:

1	• Legge til kunde	Demonstrere hvordan man legger til kunde via GUI	Tenker å bruke gridBag ettersom det visstnok er veldig fleksibelt.
2	• Slette kunde	Demonstrere hvordan man endrer kunde via GUI	Tenker å bruke gridBag ettersom det visstnok er veldig fleksibelt.
3	• Endre Kunde	Demonstrere hvordan man sletter kunde via GUI	Tenker å bruke gridBag ettersom det visstnok er veldig fleksibelt.

Implementere kart:

2	• Netbeans for å implementere kart		Ideen her var å se hvordan netbeans ville representert kartet for så å se hvilken kode som var nødvendig for så å renskrive denne. Vi var usiker i valg av karttjeneste, samt at guidene vi prøvde ikke var helt velvillige å ha med
---	------------------------------------	--	--

å gjøre.

Sprint2:

Pri:	Beskrivelse:	Demo:	Notes:
1	Tegne arkitektur	Tegne gangen i systemet	Kartet trenger postTabellen for å finne riktig adresse på kartet, så vi må legge til denne
<i>Opprette MVC mellom database og GUI:</i>			
1	• Impl. kontroller	Se på koden og se at gui og database kode ikke prater direkte sammen	Lage et skille mellom guikoden og databasekoden, og forhindre at disse prater direkte til hverandre. Enklere for andre å komme inn å endre på noe.
<i>Forbindelse mellom programmet og GUI:</i>			
1	• Legg til kundemetode	Lage en junit test	Skrive til databasen med en gang, oppdatere kundeLista
2	• Slett kunde metode	Lage en junit test	Skrive til databasen med en gang, oppdatere kundeLista
3	• Endre kunde metode	Lage en junit test	Skrive til databasen med en gang, oppdatere kundeLista
<i>Implementere rettListe:</i>			
1	• Legg til ny rettmetode	Lage en junit test	Skrive til databasen med en gang, oppdatere rettLista
2	• Slett rettmetode	Lage en junit test	Skrive til databasen med en gang, oppdatere rettLista
3	• Endre rettmetode	Lage en junit test	Skrive til databasen med en gang, oppdatere rettLista
<i>Implementere kart:</i>			
2	• Fail netbeans, begynte på QT	Vise i Eclipse	Netbeans funket dårlig, gått over til QT, lar markør på kartet ligge, og fokuserer på å få opp et kart.
<i>Implementere GUI meny:</i>			
1	• Legge til rett	Kjøre gjennom skjermbildene	Lag et nytt panel for hvert skjermbilde
2	• Slette rett	Kjøre gjennom skjermbildene	Lag et nytt panel for hvert skjermbilde
3	• Endre rett	Kjøre gjennom skjermbildene	Lag et nytt panel for hvert skjermbilde

Sprint3:

Pri:	Beskrivelse:	Demo:	Notes:
<i>Forbindelse mellom programmet og GUI:</i>			
1	• Legge til bestillignmetode	Skrive junit tester	Skrive til databasen med en gang,

			oppdatere bestillingsLista
2	• Hent bestillinger til kokk	Skrive junit tester	Sette en ferdigLaget variabel på bestillingen og sortere etter den
2	• Hent bestillinger til leveringsbud	Skrive junit tester	Sette en ferdigLevert variabel på bestillingen og sortere etter den
<i>Implementere kart:</i>			
2	• JXMapKit, yahoo placefinder	Vise kart	Bruk yahoo placefinder til å finne koordinater, bruke JXMapKit for å vise kart
2	Implementere oversikt over ikke effektuerte bestillinger	Vise ved hjelp av skjermbilder	Koble sammen hentBestillingerKokk() til gui. Bruke scrollTable
2	Implementere oversikt over bestillinger som er leveringsklare	Vise ved hjelp av skjermbilder	Koble sammen hentBestillingerPizzaBud() til gui. Bruke scrollTable
1	Implementere muligheten for å legge til bestilling	Vise ved hjelp av skjermbilder, se i databasen	Kolbe sammen addBestilling() til gui.

Sprint4:

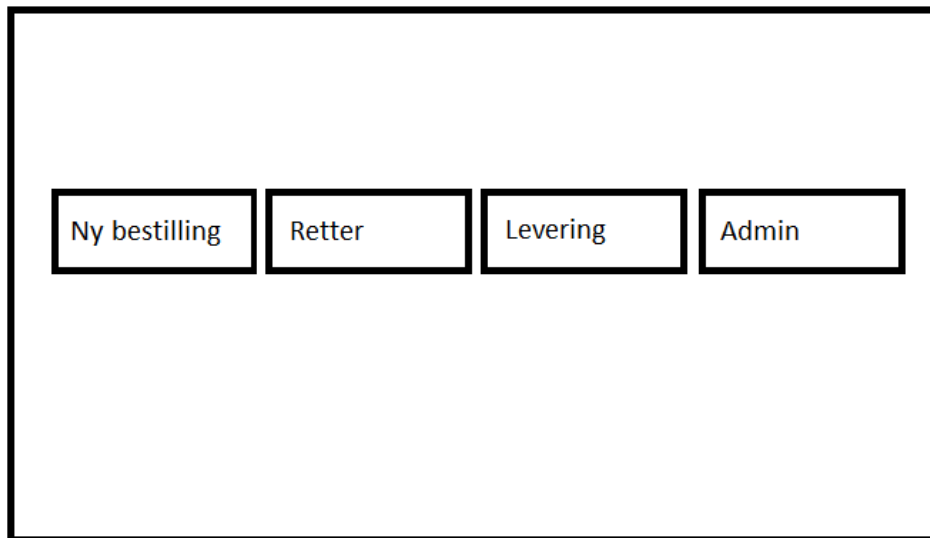
Pri:	Beskrivelse:	Demo:	Notes:
<i>Forbindelse mellom programmet og GUI:</i>			
2	• Slette bestillingmetode	jUnit test	Skrive til databasen med en gang
1	Implementere muligheten for å legge til bestilling	Vise ved hjelp av skjermbilder	Koble addBestilling() sammen med gui'et
1	Administrere fraktpriiser	jUnit tester og skjermbilder	Lage metodene og koble de sammen med gui
2	Administrere bestillinger	jUnit tester og skjermbilder	Koble bestillingsmetodene sammen med gui'et

Sprint5:

Finpuss.

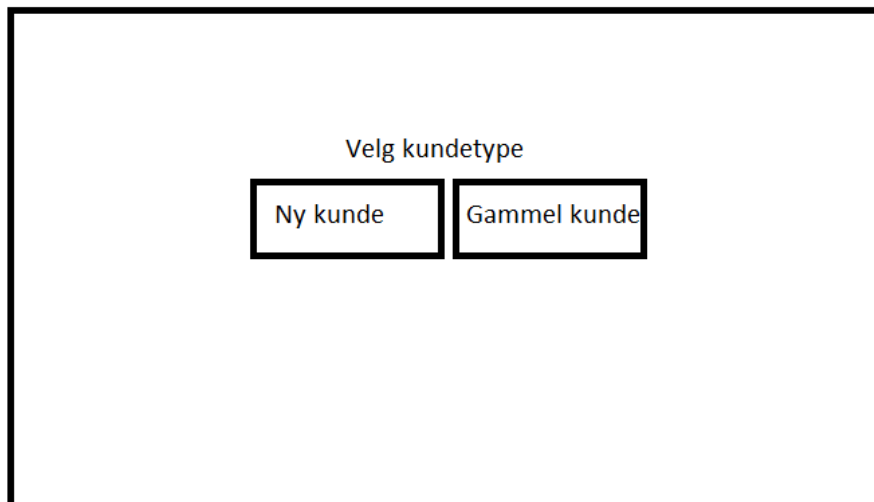
GUIskisser

Hovedside:



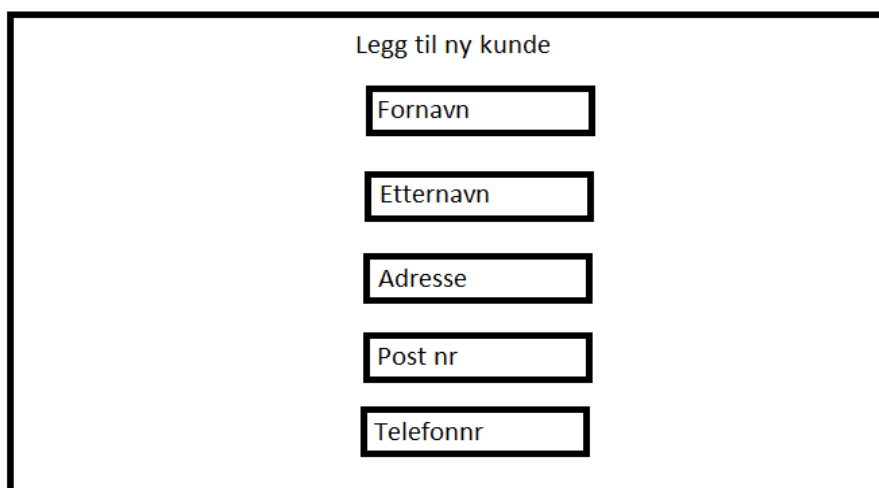
A horizontal navigation bar containing four buttons: "Ny bestilling", "Retter", "Levering", and "Admin".

Bestillingsprosessen:



Velg kundetype

Ny kunde Gammel kunde



Legg til ny kunde

Fornavn

Etternavn

Adresse


Post nr

Telefonnr

Finn eksisterende kunde

Telefon nr	Fornavn
Søk	Etternavn
	Adresse
	Postnummer
	Telefon

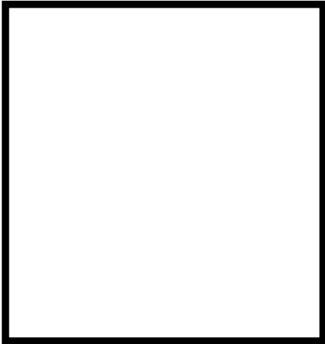
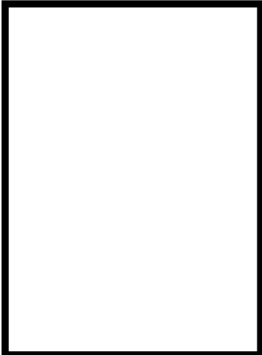
Kunde

<input type="checkbox"/> Hentes	
<input type="checkbox"/> Leveres	
<input type="checkbox"/> Kundens adresse	
<input type="checkbox"/> Leverings adresse	

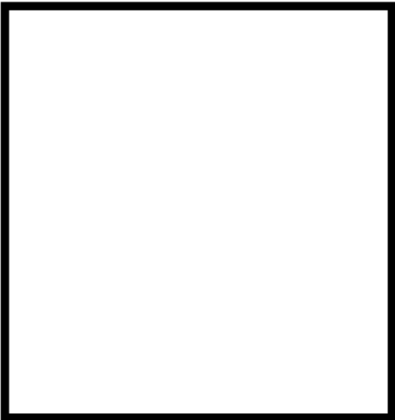
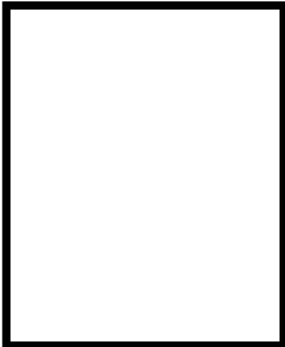
Kvittering

Kunde
Retter
pris

Kokk:

Ulagede bestillinger	Rettene
	

Leveringsbud:

Leveringsklare bestillinger	Kart
	

Admin menyen:

Admin			
Meny	Kunder	Frakt	Bestillinger

Administrer meny

Ny rett Endre meny

Endre på meny

Navn:

Beskrivelse:

Pris:

Legg til ny rett

Navn:

Pris:

Beskrivelse:

Endre fraktpriser

Fraktpris:

Endre

Grense for frifrakt:

Endre