

Super Arrangement

Mohammed Ali Al-Taryoosh, Stephan Nguyen, Fredrik Olsvold og Gytis Venckus



Prosjektgruppemedlemmer

Mohammed Ali H Al.

- Informasjonssystemer
- alnajdyhamud@gmail.com

Stephan Nguyen

- Informasjonssystemer
- Stephan-nguyen97@hotmail.no

Fredrik V. Olsvold

- Informatikk
- fredrik.olsvold@gmail.com

Gytis Venckus

- Informasjonssystemer
- gytisvenck12@gmail.com

Forord

Denne rapporten er laget som utviklingsdokument, og forklarer problemstillingen vi hadde og løsningen vi kommer med i forbindelse med faget **Software Engineering og Testing**. Prosjektgruppen ønsker å takke emne ansvarlige Mats Lindh og Lars-Erik Aabech for god veiledning og oppfølging under prosjektet.

Prosjektet gikk som forventet, hvor fordeling av arbeidet gikk rettferdig, og planen ble fulgt som planlagt. Det resulterte i et spennende og lærerikt resultat, som gruppen vil ta med seg videre i livet.

Innholdsfortegnelse

| | |
|--------------------------------|-----------|
| Prosjektgruppemedlemmer | 2 |
| Forord | 3 |
| Figurliste | 6 |
| Sammendrag | 7 |
| 1. Bakgrunn | 8 |
| 2. Kravspesifikasjon | 9 |
| 2.1 <i>Generelt</i> | 9 |
| 2.2. <i>Generelt</i> | 10 |
| 2.2.1. Funksjonelle krav | 10 |
| 2.2.2. Ikke funksjonelle krav | 11 |
| 2.3. <i>Opprette bruker</i> | 11 |
| 2.3.1. Funksjonelle krav | 11 |
| 2.3.2. Ikke funksjonelle krav | 11 |
| 2.4. <i>Bestilling</i> | 11 |
| 2.4.1. Funksjonelle krav | 11 |
| 2.4.2. Ikke funksjonelle krav | 11 |
| 2.5. <i>Betaling</i> | 11 |
| 2.5.1. Funksjonelle krav | 11 |
| 2.5.2. Ikke funksjonelle krav | 12 |
| 2.6. <i>Kvittering</i> | 12 |
| 2.6.1. Funksjonelle krav | 12 |
| 2.6.2. Ikke funksjonelle krav | 12 |
| 3. User stories | 13 |
| 4. Use-Case | 14 |
| 4.1 <i>Klient</i> | 14 |
| 4.2 <i>Produkteier</i> | 16 |
| 4.3 <i>Opprette bruker</i> | 18 |
| 4.4 <i>Betaling</i> | 20 |
| 4.5 <i>Kvittering</i> | 22 |
| 4.6 <i>Endre bestilling</i> | 24 |
| 4.7 <i>Kansellering</i> | 26 |
| 5. Sekvens diagram | 28 |
| 5.1. <i>Billettbestilling</i> | 28 |
| 5.2. <i>Godkjenne betaling</i> | 30 |
| 5.3. <i>Avbestille</i> | 31 |
| 5.4. <i>Opprette bruker</i> | 32 |
| 6. Aktivitetsdiagram | 34 |
| SE. | 4 |

| | |
|---|-----------|
| 6.1. Bestilling | 34 |
| 6.2. Godkjenne betaling | 35 |
| 6.3. Avbestilling | 36 |
| 6.4. Opprette bruker | 37 |
| 7. Prototype | 38 |
| 7.1. Veiledning for å legge til JavaFX og gson: | 38 |
| 7.1.1. Legg til library | 38 |
| 7.1.2. Gson | 39 |
| 7.1.3. JavaFX. | 40 |
| 7.2. Prototypens layout og funksjon | 44 |
| 7.2.1. Stubs. | 49 |
| 8. Testing | 51 |
| 8.1. Tester som ikke er Junit tester i prototypen | 51 |
| 8.1.1. Registrering av bruker | 51 |
| 8.1.2. Fakturering av bruker | 52 |
| 8.1.3. Bestilling av arrangement | 52 |
| 8.2. Tester som er Junit tester i prototypen | 53 |
| 8.2.1 Input tester | 53 |
| 8.2.2. Data tester | 54 |

Figurliste

| | |
|--|----|
| Figur 1: Klient..... | 15 |
| Figur 2: Produkteier..... | 17 |
| Figur 3: Opprette bruker | 19 |
| Figur 4: Betaling..... | 21 |
| Figur 5: Kvittering | 23 |
| Figur 6: Endre bestilling..... | 25 |
| Figur 7: Kansellering | 27 |
| Figur 8: Sekvens diagram billett bestilling | 29 |
| Figur 9: Sekvens diagram godkjenne betaling | 30 |
| Figur 10: Sekvens diagram avbestille ordre | 31 |
| Figur 11: Sekvens diagram opprette bruker | 33 |
| Figur 12: Prototype logg inn | 44 |
| Figur 13: Prototype Profilsidevisning feilmelding..... | 45 |
| Figur 14: Prototype registrere bruker | 46 |
| Figur 15: Prototype Profilside | 47 |
| Figur 16: Arrangement visning..... | 48 |
| Figur 17: Betaling med kort..... | 49 |
| Figur 18: Prototype Kvittering..... | 49 |

Sammendrag

Problemstillingen vi hadde i dette prosjektet går ut på å utvikle et billettsystem for de ulike lokale kinoene i regionen, der målet var at systemet skulle kunne bistå andre lokale arrangører med billettsalg og inngangskontroll. Med dette systemet, ville man forenklet hverdagen for ansatte ved de lokale kinoene i regionen.

Systemarkitekturen vil bestå av et kontrollsysteem som er bindevevet mellom de ulike delsystemer. Delsystemene består av reservasjon av billett, opprette bruker og billettkontroll.

1. Bakgrunn

Som prosjektoppgave i kurset Software Engineering & Testing, skal prosjektgruppen utforme en skriftlig rapport som tar seg av utvikling av et bedre system enn dagens kino system på vegne av den lokale kinoregionen. Gruppen har sett på nåværende løsninger og tatt inspirasjon til det gruppen mener er best for bruker og admin.

Oppgaven går ut på å utforme og utvikle et system, som skal erstatte dagens system for bestilling og kontroll av billetter til forskjellige arrangementer. Rapporten skal inneholde alle nødvendige krav, som gruppen skal sette som nødvendig, for å få til en erstatte til dagens løsning.

De viktigste delene i et slikt system mener gruppen bør være:

- Visuell side med fokus på lite tekst og mer bilde
- Enkel bestillingsmulighet
- Brukervennlighet
- Brukersikkerhet

Gruppen ønsker en enkel måte for brukere å kjøpe, bestille, avbestille, og få oversikt på lokale arrangementer. Derfor skal gruppen utforme en detaljert beskrivelse på hvordan det fullstendige systemet skal være.

2. Kravspesifikasjon

Denne kravspesifikasjonen gjelder en overordnet billett og reservasjonssystem for de lokale kinoene i regionen. Formålet med dette systemet er at systemet kan inneholde andre lokale arrangementer av andre lokale arrangører (sopptur, konserter, barnas gård etc.). Ved å lage dette systemet for underholdningsbransjen, kan man redusere mulige utfordringer og gjøre det enklere for kunder å bestille billetter til ulike arrangementer.

2.1 Generelt

| Krav: | Funksjonelle: | Ikke-funksjonelle: | Arbeidsmengde: |
|---|---------------|--------------------|----------------|
| Betale for billett | x | | Stor |
| Betale med vipps | x | | Stor |
| Regelmessig sikkerhetsoppdatering | x | | Stor |
| Responstid i nåtid | x | | Middels |
| Endre bestilling | x | | Middels |
| Hente informasjon om ledige billetter | x | | Middels |
| Data skal behandles med personvernloven | x | | Stor |
| Skal kunne lage bruker for nettsiden | x | | Middels |
| Fonten skal alltid være svart | | x | Lite |
| Skal koste en viss sum | | x | Lite |
| Responstid på 1-4 sekunder | | x | Stor |
| Verifisering av kort og betaling skal ta under 5 sekunder | | x | Stor |
| Kunden skal få kvittering av handel | x | | Stor |

| | | | |
|--|----------|----------|---------|
| Billett kontroll skal sjekke om billetten er gyldig | x | | Stor |
| Kontrollen skal vise hvilken sal og sete kunden skal gå til. | x | | Middels |
| Websiden skal kjøres på .no domene | | x | Middels |
| Navnet på nettsiden skal være Super Arrangement | | x | Middels |
| Logoen skal inneholde to billetter på kryss med tekst under | | x | Middels |
| Server skal motta forespørsler fra en liste med godkjente IP adresser i administrasjonen. | x | | Stor |
| Server skal ha som standard å nekte alle forespørsler, bare slippe gjennom hvitlistede IP-Adresser | | x | Stor |
| Server skal ha internettilgang | x | | Stor |

2.2. Generelt

2.2.1. Funksjonelle krav

- Regelmessig sikkerhetsoppdatering
- Respons i nåtid
- Data skal behandles med personvernloven

- Server skal motta forespørsler fra en liste med godkjente IP adresser i administrasjonen.
- Kontrollen skal vise hvilken sal og sete kunden skal gå til.
- Server skal ha internettilgang.

2.2.2. Ikke funksjonelle krav

- Fonten skal alltid være svart
- Responstid på 1-4 sekunder
- Websiden skal kjøres på .NO domene.
- Navnet på nettsiden skal være Super Arrangement.
- Logoen skal inneholde to billetter på kryss med tekst under.
- Server skal ha som standard å nekte alle forespørsler, bare slippe gjennom hvitlistede IP Adresser

2.3. Opprette bruker

2.3.1. Funksjonelle krav

- Skal kunne lage bruker for nettsiden
- Informasjon til bruker blir lagret i database
- Passord må valideres
- Epost kan ikke være brukt

2.3.2. Ikke funksjonelle krav

- Brukeren får bestemt tid på å lage bruker

2.4. Bestilling

2.4.1. Funksjonelle krav

- Bestillingen skal koste en viss sum
- Bruker skal ha mulighet til å endre bestilling i ettertids
- Bruker skal kunne se hvor mye produktet koster
- Bruker skal kunne bestille med sin bruker
- Billett kontroll skal sjekke om billetten er gyldig

2.4.2. Ikke funksjonelle krav

- Datoformatet som skal brukes skal være på formen "DD.MM.YYYY"

2.5. Betaling

2.5.1. Funksjonelle krav

- Betale for billett
- Betale med vipps

- Betale med bankkort
- Kan bli betalt med gavekort eller tilgodelapp

2.5.2. Ikke funksjonelle krav

- Sjekke om betaling godkjent på 3-6 sekunder
- Sørge for at HTTPS blir brukt for sikker betaling

2.6. Kvittering

2.6.1. Funksjonelle krav

- Kunde skal få kvittering av handel
- Kunde skal få på enten SMS eller E-Mail
- Kunde skal få kvittering av kansellering eller endring

2.6.2. Ikke funksjonelle krav

- Skal ta maks 2 minutter etter kjøpet er bekreftet til kunden får kvittering.
- Kvittering blir sendt med HTML kode

3. User stories

- Som bruker vil jeg kunne opprette en bruker
- Som bruker vil jeg kunne bestille forskjellige arrangementer fra samme sted
- Som bruker vil jeg kunne se mine bestillinger, og ev. Endre på dem
- Som bruker vil jeg kunne betale med vipps/kort eller betale kontant på oppmøtestedet
- Som bruker vil jeg kunne få bekreftelse på bestilling på e-post/SMS.
- Som admin vil jeg kunne se kundenes bestillinger.
- Som admin vil jeg kunne legge til arrangementer til i systemet.
- Som admin vil jeg kunne endre på bestillinger.
- Som admin vil jeg kunne kansellere bestillinger.
- Som admin vil jeg kunne bestille på vegne av en kunde.
- Som admin vil jeg kunne gi beskjed til kunder dersom et arrangement blir kansellert/endret.
- Som admin vil jeg kunne refundere manuelt dersom kunde avbestiller.

4. Use-Case

4.1 Klient

Navn: Klient

Use-Case: US1

Beskrivelse:

Dette viser hva brukeren kan bruke systemet til.

Utløser: Skrive inn nettsideadresse

Pre-betingelser:

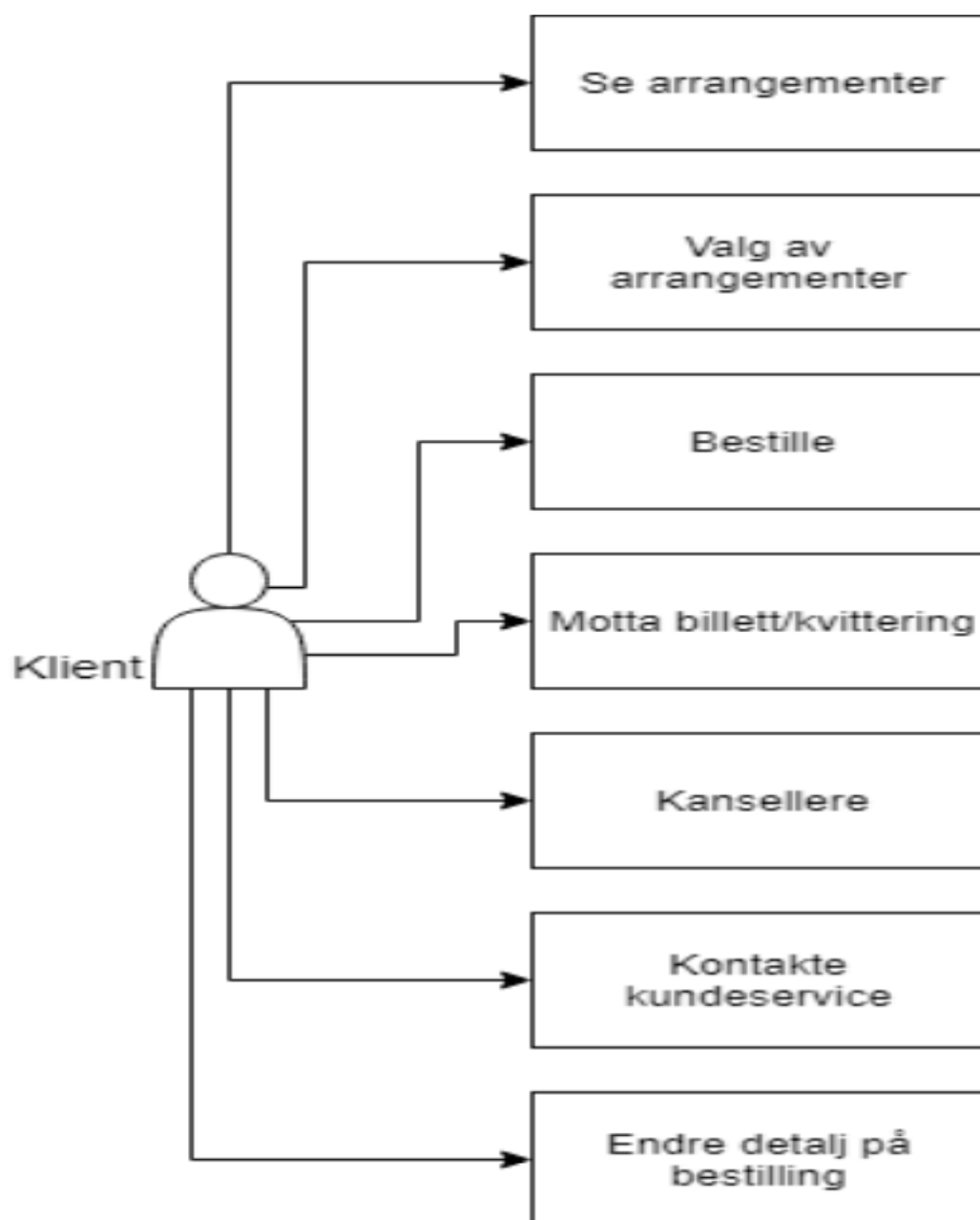
- Har internett
- Bruker en godkjent IP-adresse
- Kunde har skrevet inn riktig adresse

Post-betingelser:

- Velge arrangement for bestilling
- Kan bestille
- Få kvittering
- Kontakte kundeservice
- Kansellere
- Endre billett detalj

Normal hendelsesflyt:

1. En bruker åpner websiden på en valgt nettleser.
2. Server får informasjon om tidspunkt og dato av database etter spørring
3. Server sender HTML dokument til klient slik at de får sett nettsiden.
4. System viser brukeren sanntidsinformasjon om arrangementer og ledige plasser.
5. Bruker skal kunne velge arrangement, seter, dato og tid.
6. Bruker blir sendt til forskjellige nye HTML sider, avhengig av handling. F.eks.: bestille, opprette bruker, kanselering.
7. Use-case Slutter



FIGUR 1: KLIENT

4.2 Produkteier

Navn: Produkteier

Use-Case: US2

Aktør: Produkteier eller ansatt

Beskrivelse:

Hva produkteier kan gjøre med en Administrator bruker. Brukeren må være innlogget med admin bruker for å utføre disse handlingene.

Utløser: Når produkteier eller ansatt logger inn med administratorbruker og skal gjøre det en vanlig bruker ikke kan.

Pre-betingelser:

- Være innlogget med administratorbruker
- Ha tilgang til internett
- Bruker en godkjent IP-adresse

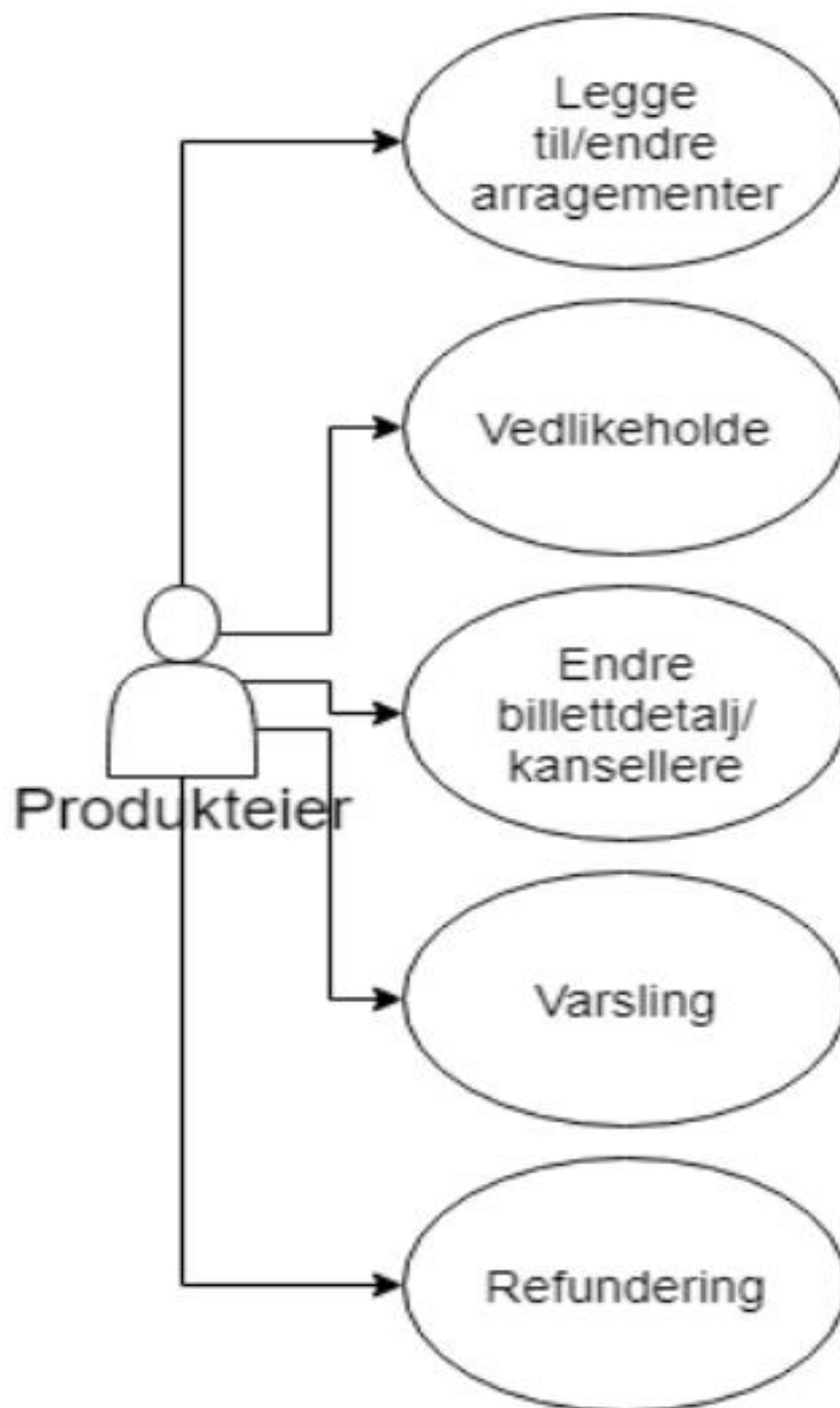
Post-betingelser:

- Administrator skal kunne endre bestilling.
- Skal kunne legge til eller endre arrangementer som skal vises på nettsiden.
- Kunne foreta vedlikehold av systemet.
- Skal kunne hjelpe kunde med refundering av bestilling om kunde trenger hjelp til dette
- Send ut personlig varsel ang bestilling om det er behov.

Normal hendelsesflyt:

1. Personen/ansatte skal kunne logge inn med administratorbruker
2. Serveren kontakter databasen og foretar en spørring med tidspunkt og dagens dato
3. Systemet sender et html-dokument til brukeren der det blir vist hva vedkommende kan gjøre som innlogget med administratorbruker
4. Administratorbruker vil da få en oversikt over ulike muligheter personen har. I dette tilfelle vil det bli vist fanen "Legge til/endre arrangementer, vedlikehold, endre billettdetalj/kansellere, varsling eller refundering."
5. Stegene deretter vil da avhenge av hva brukeren velger å gjøre.
6. Use-Case slutter

Variasjoner:



FIGUR 2: PRODUKTEIER

4.3 Opprette bruker

Navn: Opprette bruker

Use-Case: US3

Aktør: Kunde

Beskrivelse:

Denne Use-Casen beskriver hva kunden skal gjøre når kunden skal opprette en bruker i systemet. Alle "boksene" må være fylt ut for å kunne opprette en bruker.

Utløser: Når kunden trykker på "opprette bruker" knappen vil denne Use-casen være til bruk.

Pre-betingelser:

- Brukeren må ha nett tilgang
- Bruker en godkjent IP-adresse

Post-betingelser:

- Kunden skal kunne lage en bruker for dette systemet.
- Senere skal kunden kunne bestille med denne brukeren.

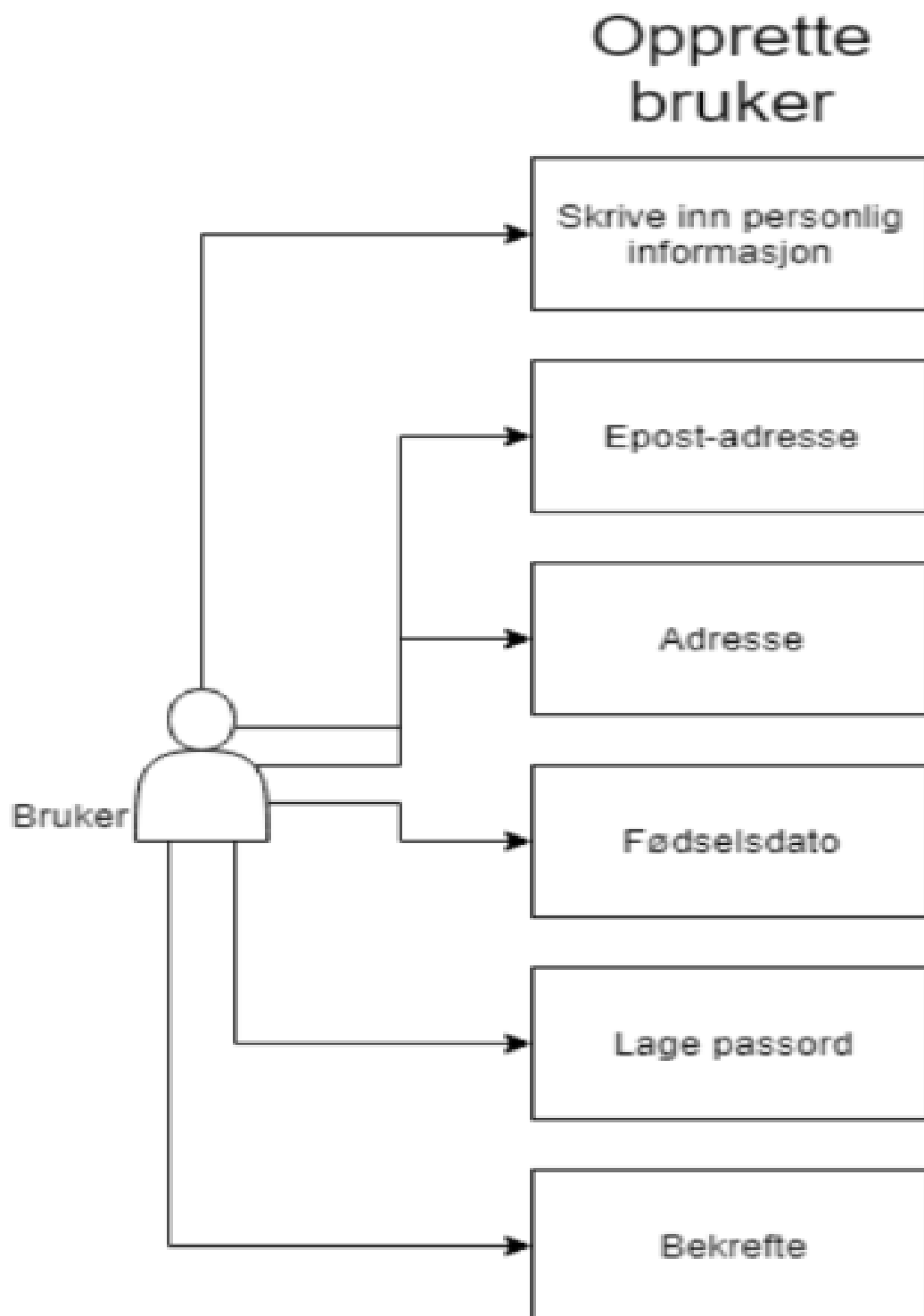
Normal hendelsesflyt:

1. Bruker trykket på "opprett bruker" knappen
2. Server kontakter database for å spørre om tidspunkt og dato.
3. Serveren gir brukeren HTML-siden.
4. Brukeren vil da få opp en nettside med oversikt over hvordan man oppretter bruker.
5. Brukeren må først skrive inn personlig informasjon
6. Skrive inn Epost-adresse
7. Skrive inn bosted/adresse
8. Skrive inn fødselsdato
9. Lage et passord med noen krav. (passordet skal ha både (Navn kan bare inneholde store og små bokstaver og mellomrom. Email kan bare inneholde store og små bokstaver, tall og tegnene _!#\$%&'*/+=?`{|}~^.- er lovlige før @. Alfakrøll er forventet og domenet på slutten må inneholde minst to tegn og maks fire. Passord må inneholde minst 8 tegn, minst en stor bokstav og minst ett tall.)
10. Verifisere om kravene er godkjent ved bekreftelse.
11. Use-case Slutter

Variasjoner:

Bruker trykker ikke "bekrefte" knappen

1. Etter 15 minutter uten å ha lagd en bruker, vil brukeren få varsel om å fullføre snares.
2. Etter nye 15 minutter vil brukeren bli kastet ut av "opprette bruker" siden og bli sendt tilbake til hovedsiden.



FIGUR 3: OPPRETTE BRUKER

4.4 Betaling

Navn: Betaling

Use-Case: US4

Aktør: Kunde

Beskrivelse:

Denne fasen beskriver hva brukeren må gjøre eller kan gjøre når vedkommende skal betale for valgt produkt/produkter

Utløser: Kunden må ha valgt et produkt eller produkter vedkommende vil betale for. Dette kan være av flere ulike sjangere (Kino og sopptur f.eks.)

Pre-betingelser:

- Billetten de skal betale for må være ledig (Sete skal ikke være tatt og arrangementet skal ha ledig plasser.)
- Bruker skal ha trykket på "gå til betaling" knapp
- Bruke en IP-adresse som ikke er sperret av serveren
- Ha tilgang til internett

Post-betingelser:

- Skal kunne fullføre kjøpet
- Ha mulighet for å velge betalingsmetode.
- Trykke godkjenn og få kjøp bekreftet.

Normal hendelsesflyt:

1. Bruker får en oversikt over hvilke metoder det finnes.
2. Bruker får velge hvilken betalingsmetode de vil foreta kjøpet med.
3. Resterende varierer fra hva kunden velger
4. Kunden trykker godkjenn og blir da sjekket om betalingen blir godkjent.
5. Use-case Slutter

Variasjoner:

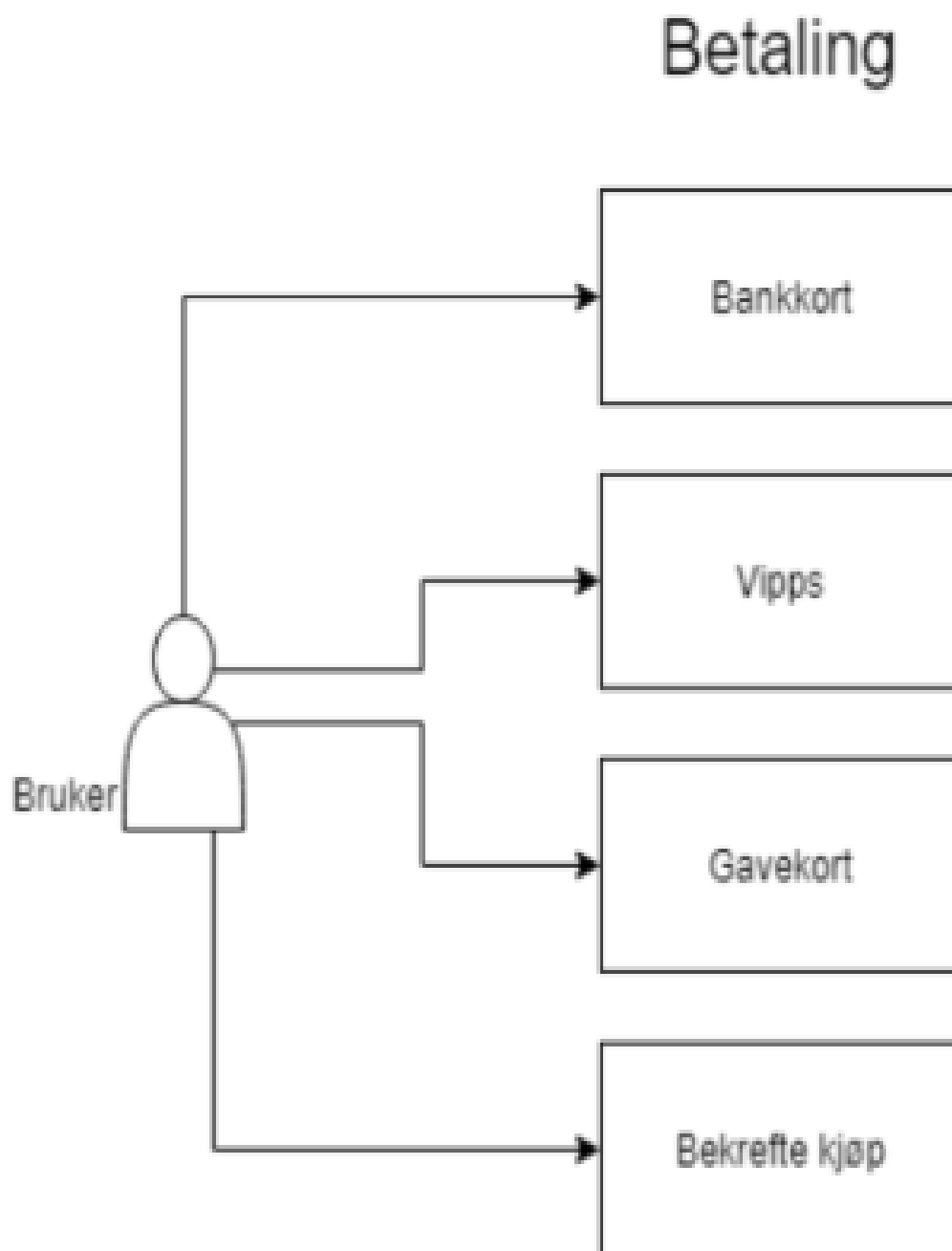
Kunden fullfører ikke kjøpet med gitt tid:

1. Kunden får varsel om at kunden har brukt 15 minutter og ber vedkommende om å fullføre, ellers blir kjøpet kansellert.
2. Etter nye 15 minutter vil kunden bli kastet tilbake til hovedsiden med varsel om å gå tilbake til betaling.

Betalingsmetoden ble ikke godkjent:

1. Kunden får ikke godkjent betaling. Får da varsel om å bytte betalingsmetode eller prøve på nytt.

2. Går tilbake til normal hendelsesflyt.



FIGUR 4: BETALING

4.5 Kvittering

Navn: Kvittering

Use-Case: US5

Aktør: System

Beskrivelse:

Beskriver hvordan serveren skal sende ut bekreftelsesmelding/Kvittering enten på SMS eller email.

Utløser: Bruker skal trykket godkjent på betaling og handel skal ha gått uten problemer.

Pre-betingelser:

- Ha fått forespørsel om hvor kunden vil få kvittering. (SMS eller Email)
- Ha tilgang til internett
- Utført et handel og fått betaling godkjent.
- Bruke en IP-adresse som ikke er sperret av serveren

Post-betingelser:

- Kvittering/bekreftelse skal ha blitt sendt ut enten på Email eller SMS til kunde.

Normal hendelsesflyt:

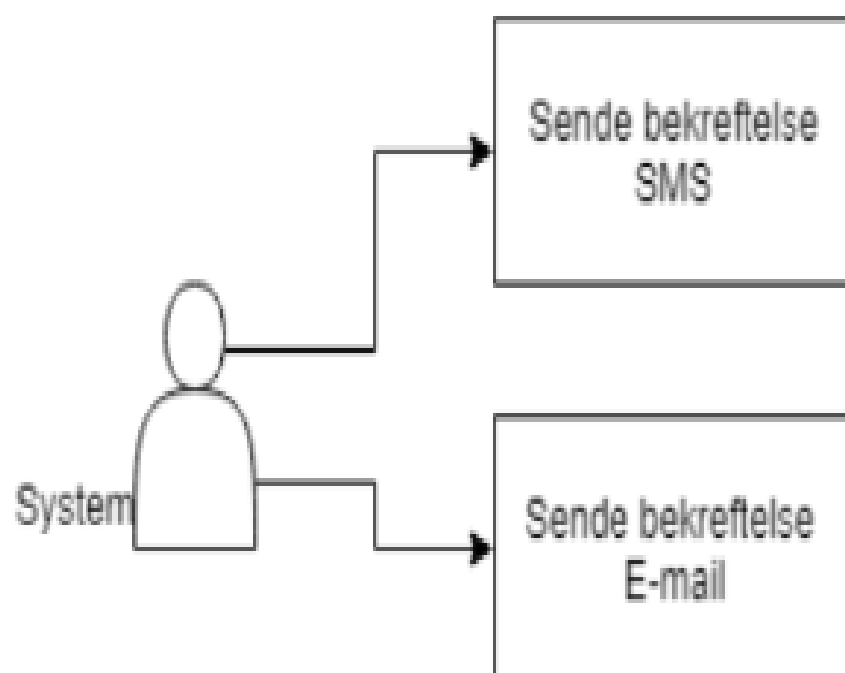
1. Systemet sender ut forespørsel til serveren om de kan sende en bekreftelse til en gitt bruker (altså kunden har betalt)
2. Serveren foretar seg jobben og prosesserer forespørselen fra systemet og sender ut SMS eller Email til angitt bruker. Enten om det er SMS eller Email avhenger av hva bruker ønsker.
3. Use-Case Slutter

Variasjoner:

Om kunde er innlogget.

1. Få automatisk kvittering på både SMS og Email.

Kvittering



FIGUR 5: KVITTERING

4.6 Endre bestilling

Navn: Endre bestilling

Use-Case: US6

Beskrivelse:

Viser hva kunden skal gjøre om vedkommende har lyst til å foreta seg en endring på sine bestillinger.

Utløser: Kunde har noe å endre og trykker på knappen "endre bestillinger" inne på profil.

Pre-betingelser:

- Kunde må ha foretatt seg bestillinger slik at de har mulighet til å på endre noe
- Må være innlogget
- Bruke en IP-adresse som ikke er sperret av serveren
- Ha tilgang til internett

Post-betingelser:

- Kunde skal kunne foreta en endring i sin bestilling. Enten om det er dato/tid, seter eller sted.

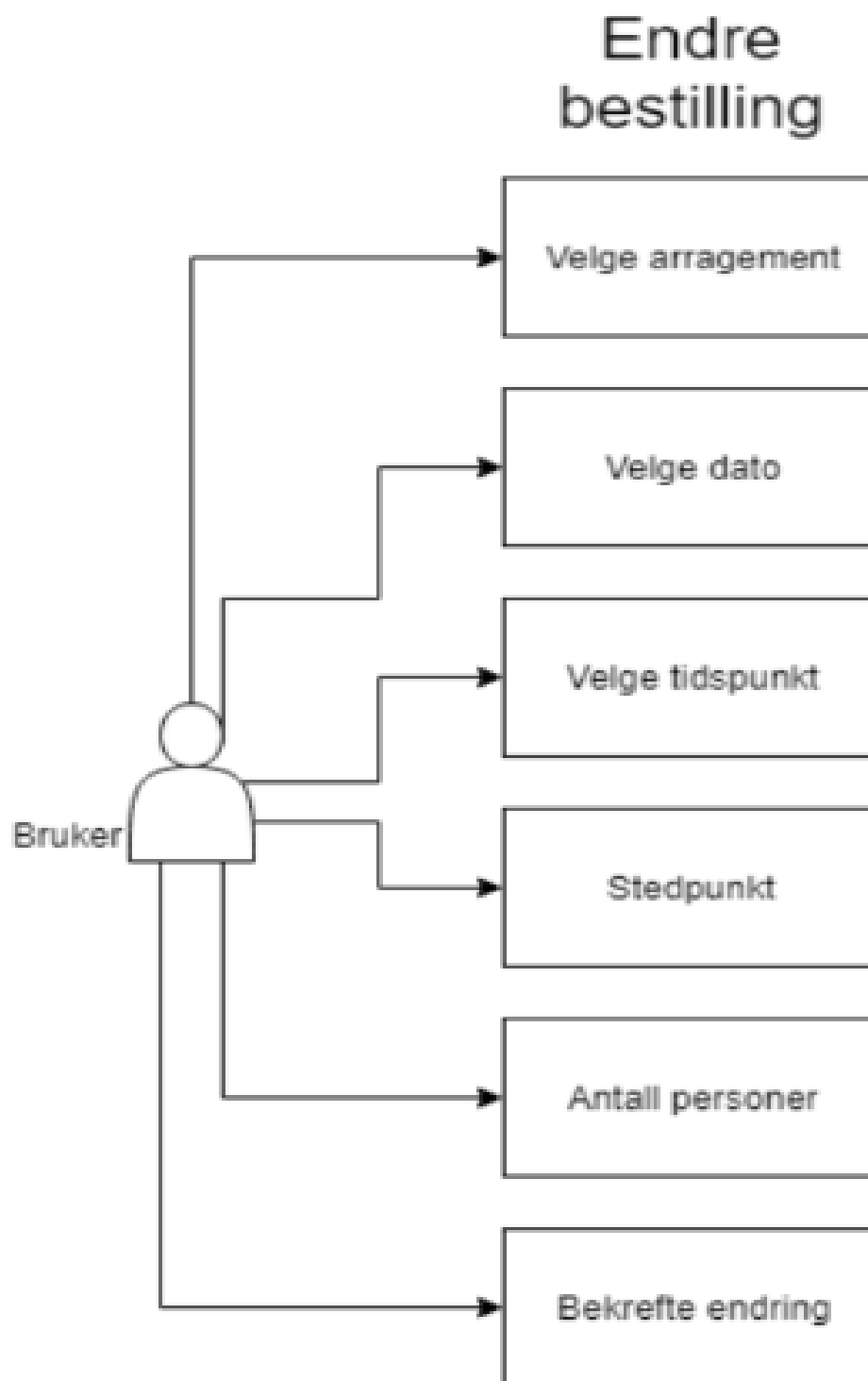
Normal hendelsesflyt:

1. Kunden skal få en oversikt over sine bestillinger frem i tid.
2. Kunden skal velge en av bestillingene i listen. Velge den de vil endre.
3. Kunden kan da velge å endre på forskjellige ting (Tid, sted, seter).
4. Kunden skal velge hvor de vil motta en kvittering av endring.
5. Kunden trykker da på "Bekreft endringer"
6. Server sender endret informasjon inn til databasen.
7. Server sender bekreftelsesmelding/kvittering på valgt plattform.
8. Use-Case Slutter

Variasjoner:

Kunden har ingen bestillinger

1. Får et varsel der det står at vedkommende ikke har noen bestillinger de kan foreta en endring på.
2. Use-case Slutter



FIGUR 6: ENDRE BESTILLING

4.7 Kansellering

Navn: Kansellering

Use-Case: US7

Beskrivelse:

Dette sier hvordan kansellerings prosess foregår for en kunde.

Utløser: Kunder ønsker å kansellere kjøp og trykker på knappen "Avbestill bestilling"

Pre-betingelser:

- Må ha utført noen bestillinger
- Må ha tilgang til internett
- Må være innlogget
- Bruke en IP-adresse som ikke er sperret av serveren

Post-betingelser:

- Skal kunne avbestille en eller flere bestillinger kunden har foretatt.

Normal hendelsesflyt:

1. Kunden skal få en oversikt over sine bestillinger frem i tid.
2. Kunden skal velge på en av bestillingene i listen. Velge den de vil kansellere.
3. Kunden skal fylle ut begrunnelse for kansellering av vare.
4. Kunden skal velge hvor de vil få bekreftelse/kvittering for kansellering.
5. Kunden skal trykke bekreft.
6. Use-case Slutter

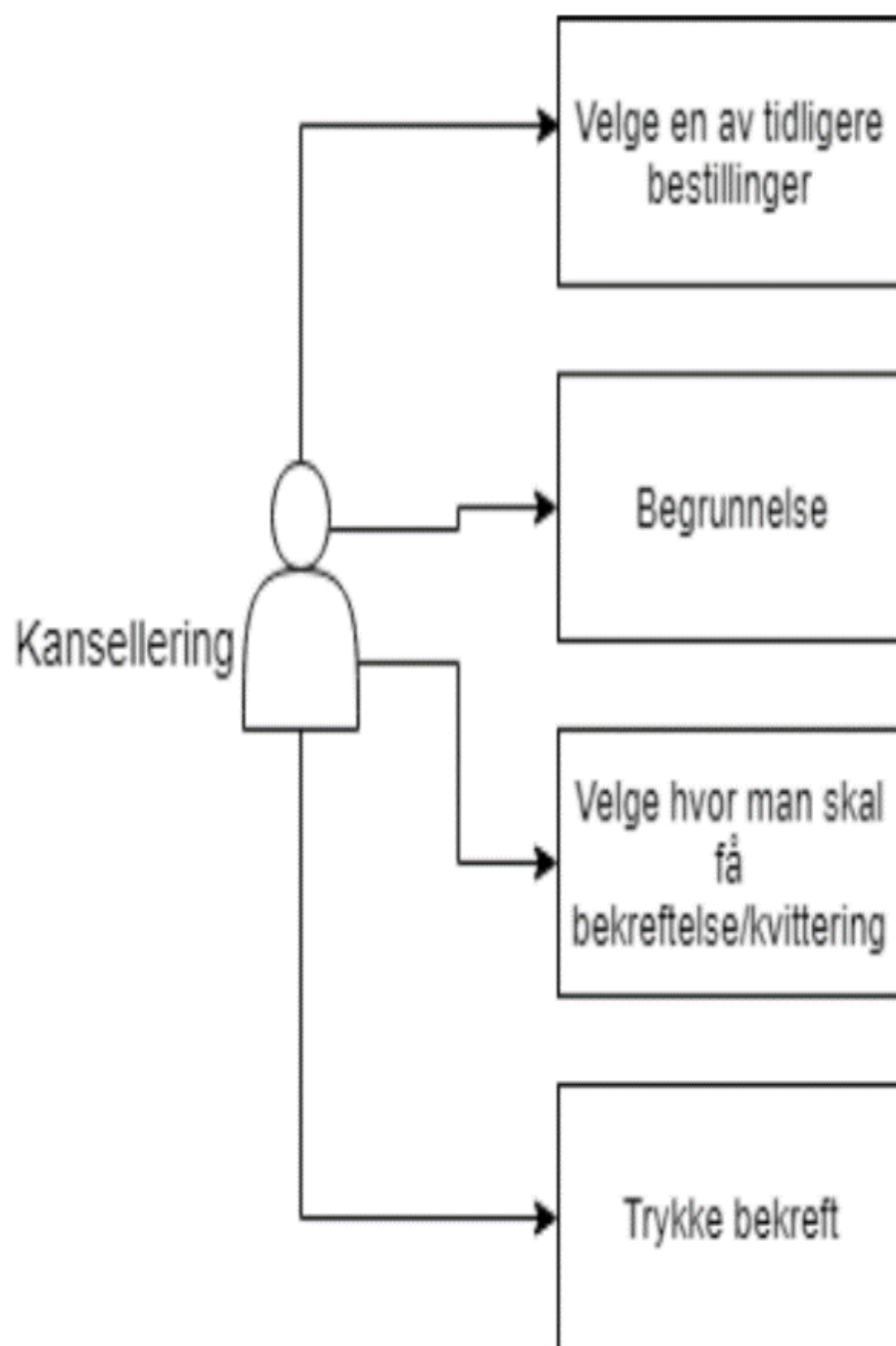
Variasjoner:

Tilgodelapp eller full refusjon?

Om kunden skal få en tilgodelapp eller full refusjon avhenger av når du foretar deg kanselleringen. Siden nettsiden ikke har reklamert at den har åpent kjøp, vil ikke alle kanselleringer føre til full refusjon.

Gruppen har valgt å si at alle kanselleringer skal resultere til at kunden skal få en tilgodelapp, med mindre kanselleringen blir foretatt innen 1 time etter bestilling, eller 3 dager før arrangementet skal starte.

Tilgodelappen vil vare i et 1 år, men ved full refusjon vil kunden bli tilbakebetalt.



FIGUR 7: KANSELLERING

5. Sekvens diagram

For dette prosjektet har vi valgt å bruke sekvensdiagram for å representere samspillet mellom ulike objekter over et bestemt tidsrom. Samtidig som at diagrammet viser oss hvilket objekt som har kontrollen til enhver tid. Det spesifiserer nøyaktig hva som skjer når.

Vi synes at bestillings, avbestillings, oppretting av ny bruker og godkjenning av betaling er noe av de mest viktige prosesser, og derfor valgte å lage sekvensdiagram for disse prosesser.

5.1. Billettbestilling

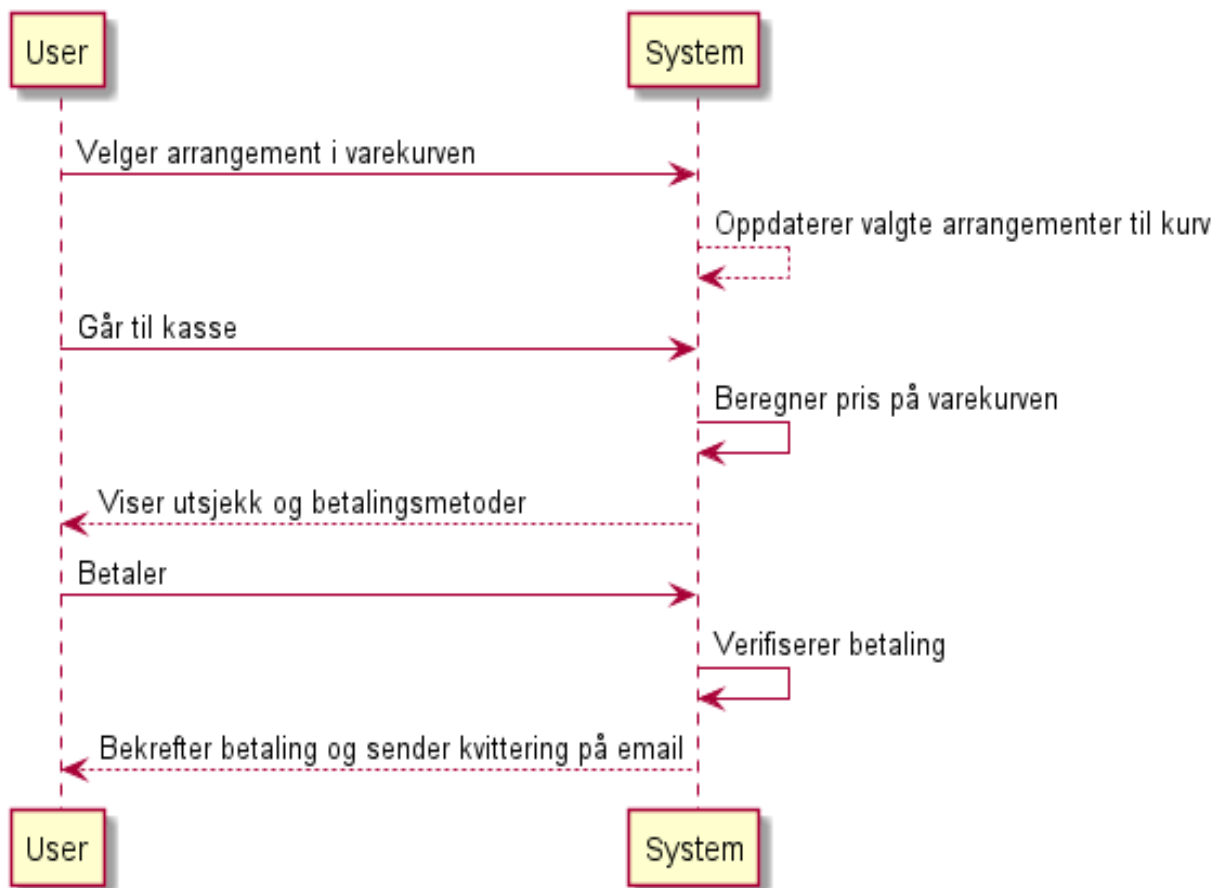
Navn: Billettbestilling

Sekvensdiagram: SD1

Beskrivelse:

Figuren 8 viser hva som foregår når en kunde skal foreta seg et kjøp. Når brukeren er logget inn kan den gå inn og velge arrangementer den vil kjøpe billett til.

Systemet legger til det valgte arrangementet i en varekurv. Når brukeren er ferdig med å legge til varer i varekurven velger brukeren å gå til kassen. Systemet beregner pris og viser varekurven og venter på betaling fra brukeren. Etter brukeren har betalt sendes en email med bekreftelse og kvittering.



FIGUR 8: SEKVENSDIAGRAM BILLETT BESTILLING

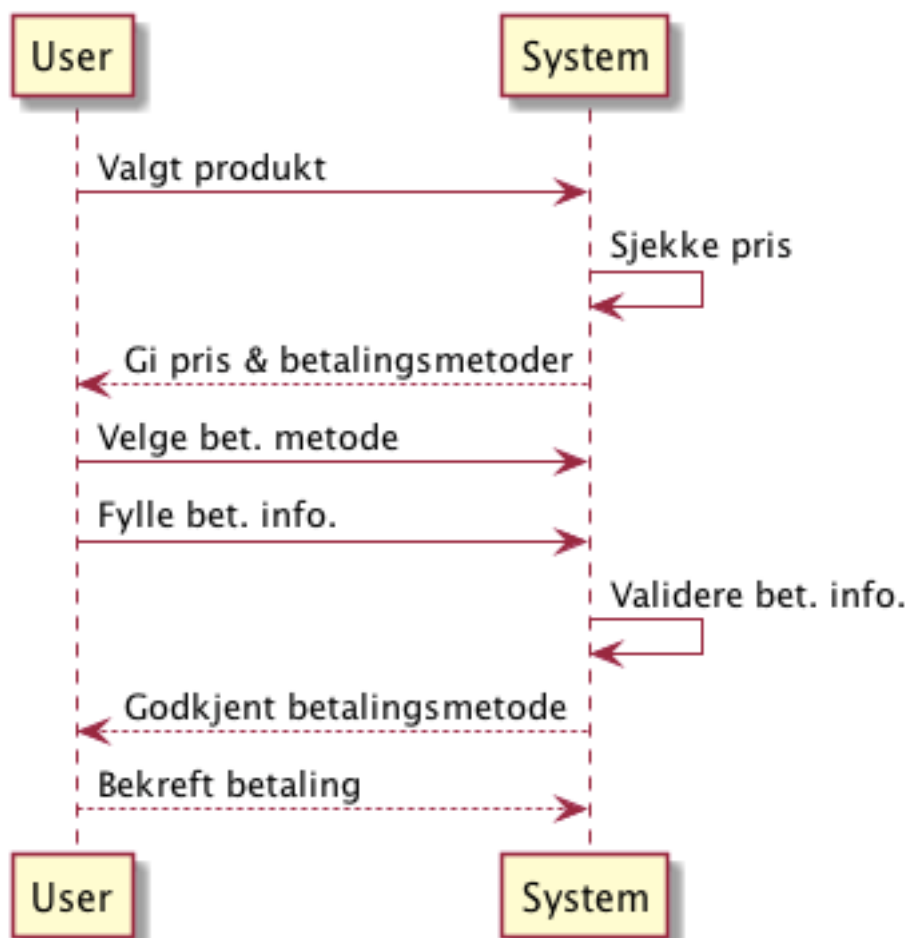
5.2. Godkjenne betaling

Navn: Godkjenne betaling

Sekvensdiagram: SD2

Beskrivelse:

Figuren 9 viser til hvordan en betaling/handel blir godkjent. Først av alt, må kunden ha valgt noen produkter, før betalingen kan bli godkjent. Deretter sjekker systemet hva valgt produkt koster. Dette kan bli gjort med flere varer samtidig. Systemet legger bare til prisene og gir den til kunden. Systemet gir samtidig brukeren flere betalingsmetoder. Siden systemet vårt kan ta betalt med både vipps, kort og tilgodelappkode. Kunden skal deretter velge betalingsmetoden de vil betale med, og fylle gitte felter. Dette er alt fra kortnummer, navn, telefonnummer, tilgodelapp kode, spørsmål tilgodelappkode. Spørsmål hva kunden vil betale med. Når kunden har fylt inn disse feltene, skal systemet validere betalingen. Når alt er godkjent, vil systemet be brukeren om å godkjenne kjøpet.



FIGUR 9: SEKVENSDIAGRAM GODKJENNE BETALING

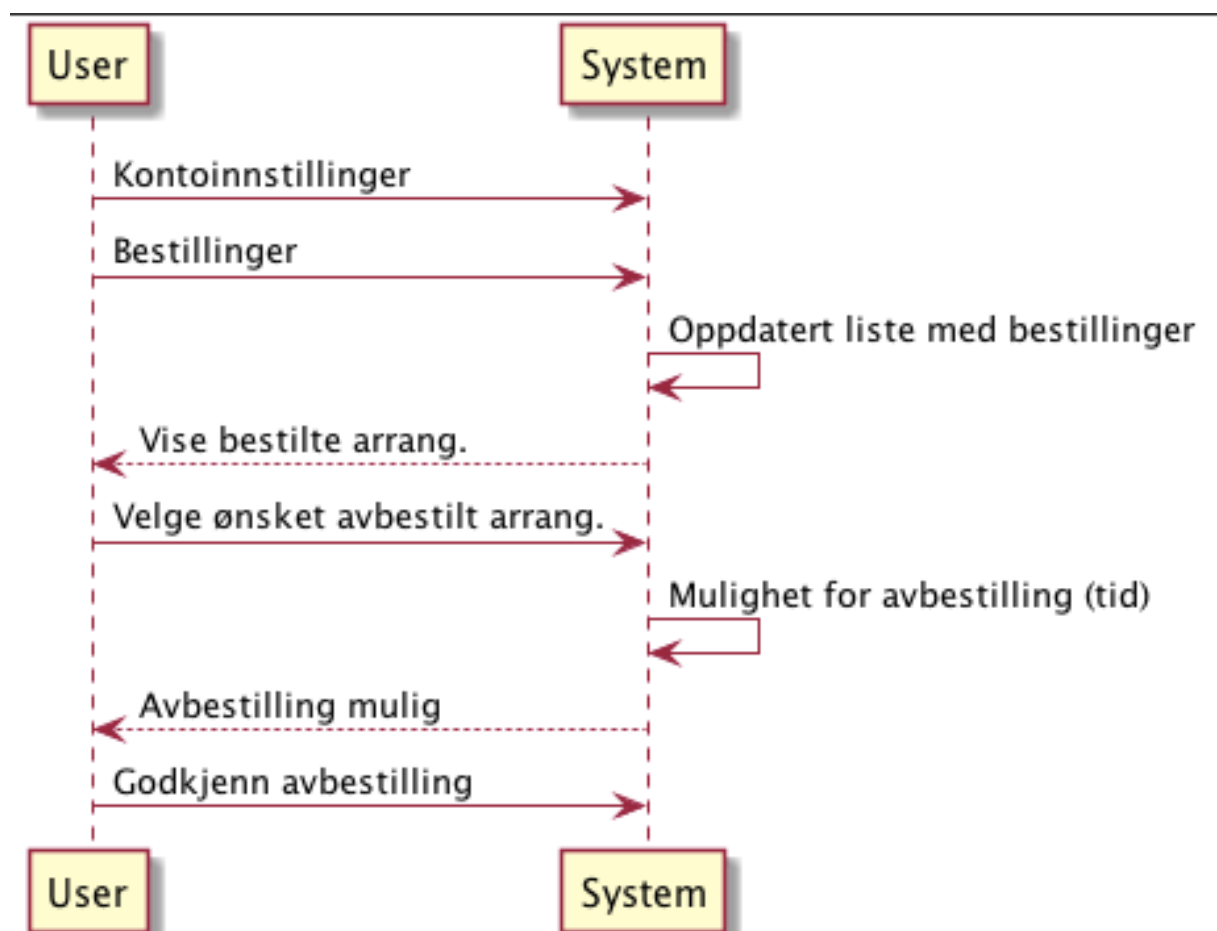
5.3. Avbestille

Navn: Avbestille

Sekvensdiagram: SD3

Beskrivelse:

Figuren 10 viser til hvordan prosessen er når en kunde skal avbestille en billett. Før alt dette kan skje, må brukeren allerede være innlogget slik at systemet kan sjekke hva vedkommende har bestilt i fortiden. Om kunden er allerede innlogget, skal personen først gå inn i kontoinnstillinger, der får personen flere valgmuligheter, men skal trykke på bestillinger. Systemet vil da sjekke tidligere bestillinger, og oppdatere denne listen slik at kunden får riktig liste med sine bestillinger. Systemet vil da gi kunden en oversikt over sine bestillinger. Kunden velger dermed en bestilling de vil avbestille. Systemet sjekker om bestillingen kan bli avbestilt (Om kunden får full refusjon eller bare tilgodelapp med tanke 3 dagers og innen 1 time regelen.) Etter det, vil systemet gi beskjed til kunden om det er mulig med full refusjon eller bare tilgodelapp. Kunden må deretter godkjenne avbestilling.



FIGUR 10: SEKVENSDIAGRAM AVBESTILLE ORDRE

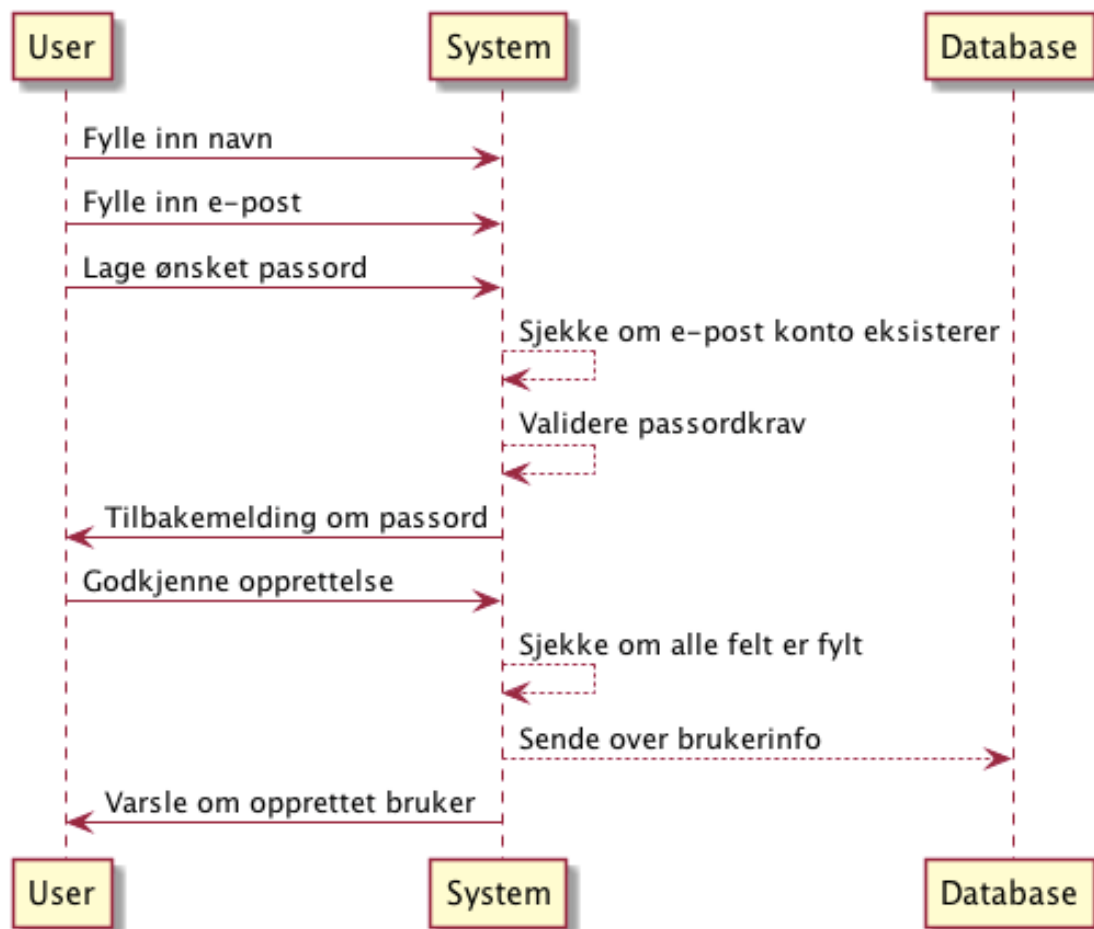
5.4. Opprette bruker

Navn: Opprette bruker

Sekvensdiagram: SD4

Beskrivelse:

Figuren 11 viser til prosessen til bruker, system og databasen når en kunde skal opprette en bruker. Dette diagrammet viser til handlinger som skjer når brukeren allerede er inni "opprette bruker" siden. Når kunden er inni siden, vil siden inneholde tomme felter, som kunden skal fylle ut. Dette er navn, e-post og ønsket passord. Når kunden gir disse informasjonene til systemet, skal systemet sjekke om e-posten er brukt og om passordet validerer kravene. Om alt er bra, vil systemet gi beskjed om at kunden kan godkjenne opprettelse, om kravene ikke er oppfylt, må kunden prøve på nytt. Etter kunden har godkjent opprettelsen, vil systemet sjekke om alle feltene er fylt inn. Om det er tilfellet, vil systemet sende over brukerinfo til databasen. Om dette går uten problemer, vil systemet gi beskjed til kunden at vedkommende har opprettet bruker.



FIGUR 11: SEKVENSDIAGRAM OPPRETTE BRUKER

6. Aktivitetsdiagram

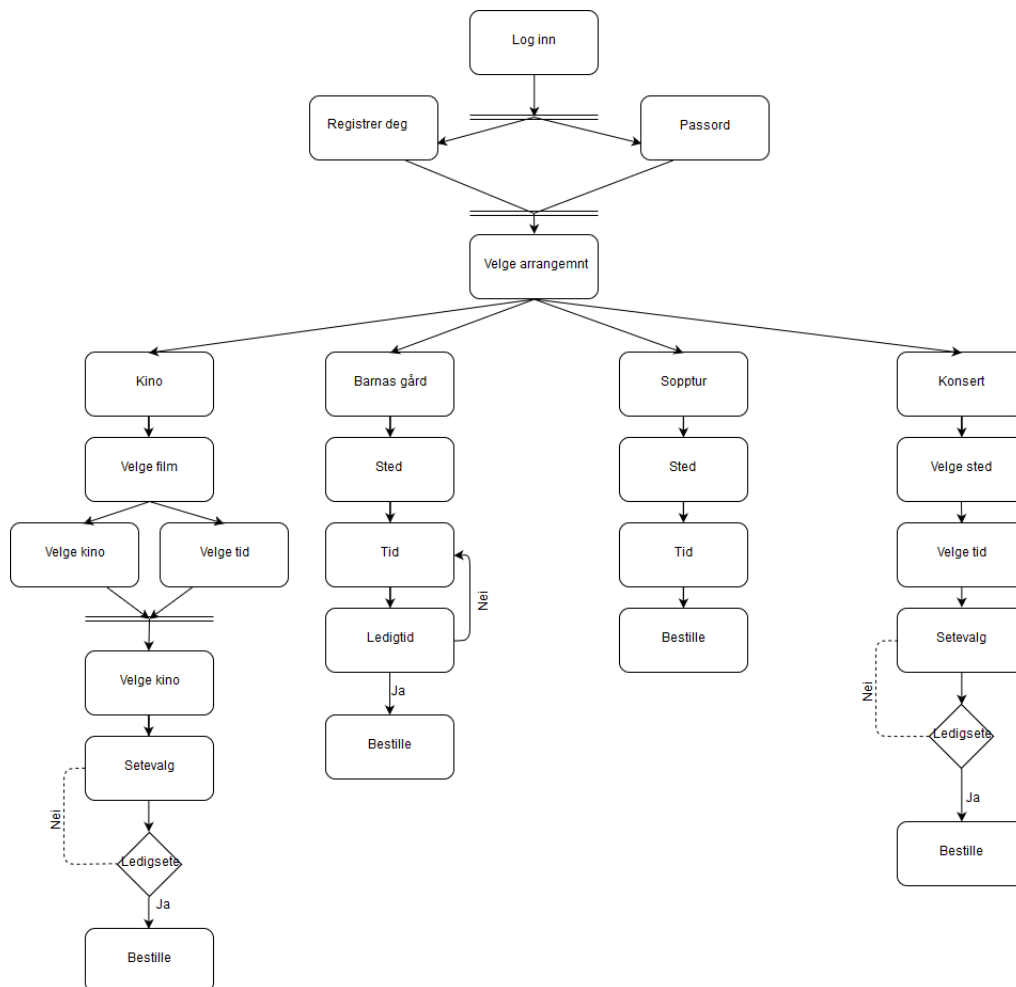
Aktivitetsdiagram er en enkelt illustrasjons form som viser hva som skjer i en arbeidsflyt. Ut av aktivitetsdiagram ser vi hvilke aktiviteter som kan utføres parallelt og hvilke alternative veier det finnes i arbeidsflyten.

Vi synes at bestilling, avbestilling, oppretting av ny bruker og godkjenning av betaling er noe av de mest viktige prosesser, og derfor valgte og lage aktivitetsdiagrammer om disse prosessene.

6.1. Bestilling

Dette aktivitetsdiagrammet visualiser aktivitetsflyt i en bestillingsprosess. For å bestille det kundene vil, må de gå gjennom flere steg i prosessen. Først og fremst, som vist i diagrammet, må kunden logge inn eller registrere bruker. Når kunden er logget inn og har valgt arrangement må de gjennom flere steg som for eksempel valg av sted, tid, sete osv.

Etter at nødvendig informasjon er fylt ut og det ikke blir vist noe feilmelding er bestillingen akseptert og neste steg er betalingsprosessen.



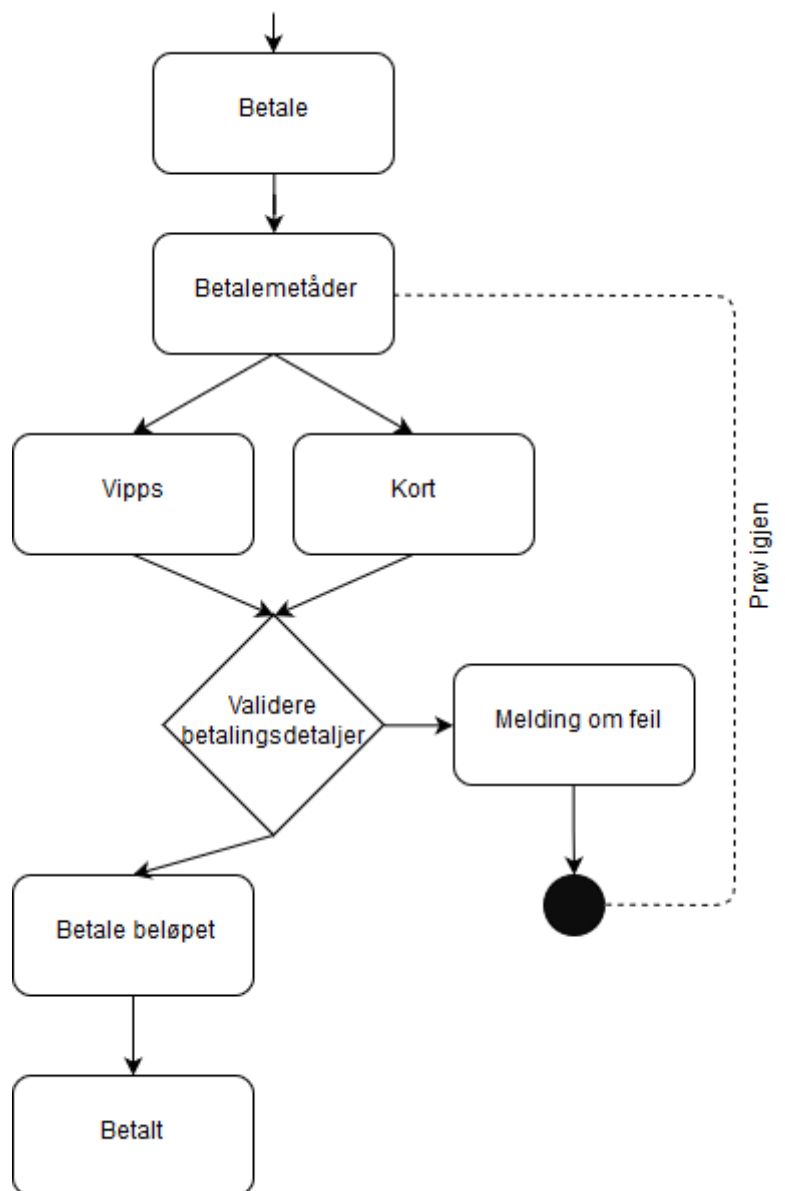
6.2. Godkjenne betaling

Kunden kan velge hva slags betalingsmetode som passer best. Man kan velge kort eller betale med Vipps. Hvis kunden velger å betale med kort, må man fylle inn kortinformasjon, etter det sender systemet ut forespørsel til banken for å bekrefte at kortinformasjon er riktig, og om kortinnehaveren har nok dekning for å fullføre kjøpet.

Hvis kunden har Vipps og velger å betale med den, alt må gjøres er å skrive inn telefonnummer og akseptere betaling via Vipps-appen på telefon. Deretter skjer prosessen som sjekker om det er nok dekning på konto for å fullføre kjøpet.

Verifisering gjennom kjøpmann og bank skjer raskt og sikkert. Etter at kunden har betalt vises en «Takk» melding og at kjøpet er gjennomført.

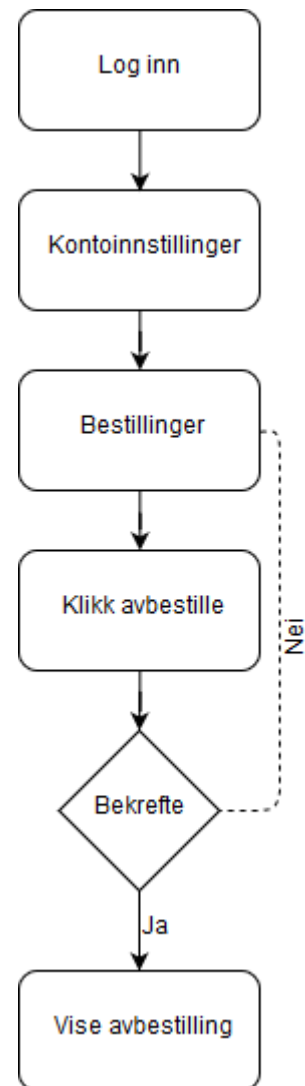
Hvis av en eller annen grunn kunden får feilmelding om at betaling ikke er fullført, blir kunden bedt om å velge en annen betalingsmetode eller prøve igjen.



6.3. Avbestilling

Hvis avbestilling er tilgjengelig for bestemt aktivitet, må det være lett tilgjengelig for kunden. Alt kunden må gjøre er å logge inn og under kontoinnstillinger finnes det en bestillingshistorikk hvor man kan avbestille et bestemt arrangement. Det kan presiseres at det ikke alltid er mulig å avbestille billetter.

Kunden kan ikke bli refundert hvis det er mer enn en time etter bestilling eller mindre enn tre dager før selve arrangementet starter. Årsaken til dette blir at arrangører skal ha en mulighet å markedsføre de siste billettene som ble avbestilt.



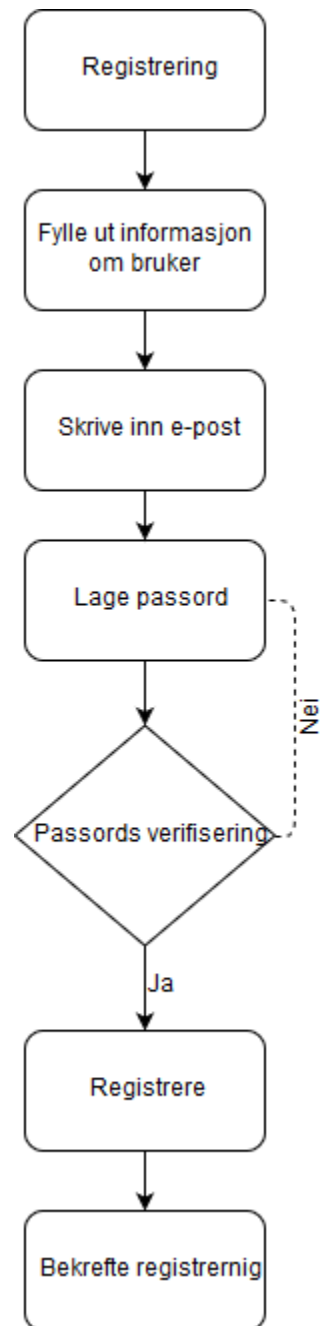
6.4. Opprette bruker

For å ha mulighet til å kjøpe billetter på nettsiden, må kunden være registrert bruker. Alle kunder kan registreres lett etter at de har fylt ut informasjon om seg selv, som for eksempel navn, fødselsdato, poststed, passord og e-post.

Når kunden skal lage passord, blir det vist informasjon om passordkrav. Hvis kunden får feilmelding på passord, må passordkrav oppfylles.

Kunden blir sendt en e-post med e-post bekreftelse.

Deretter får man tilgang til informasjon, bestillinger og privilegier utilgjengelig for ikke-registrerte brukere, vanligvis referert til som gjester.



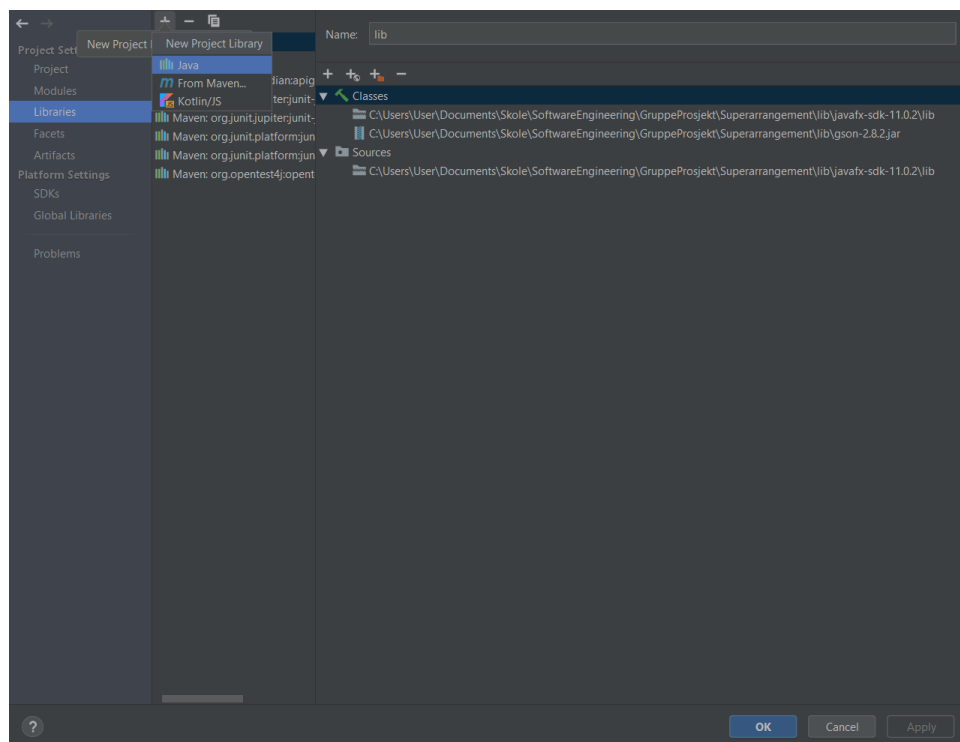
7. Prototype

Prototypens grensesnitt er laget med JavaFX. Det er et enkelt verktøy for å lage GUI. for desktoper og nettleisere på Windows, Linux og MacOS. JavaFX Prosjektet er skrevet i Java 12 og har derfor ikke JavaFX innebygd og må legges til fra ekstern kilde. Filene ligger i mappen \lib\ i prosjektet og må legges til som library i project structure. I tillegg bruker vi gson for å serialisere og deserialisere objekter til Json. Dette må også legges til fra ekstern kilde og ligger også klar for implementasjon i prosjektmappen.

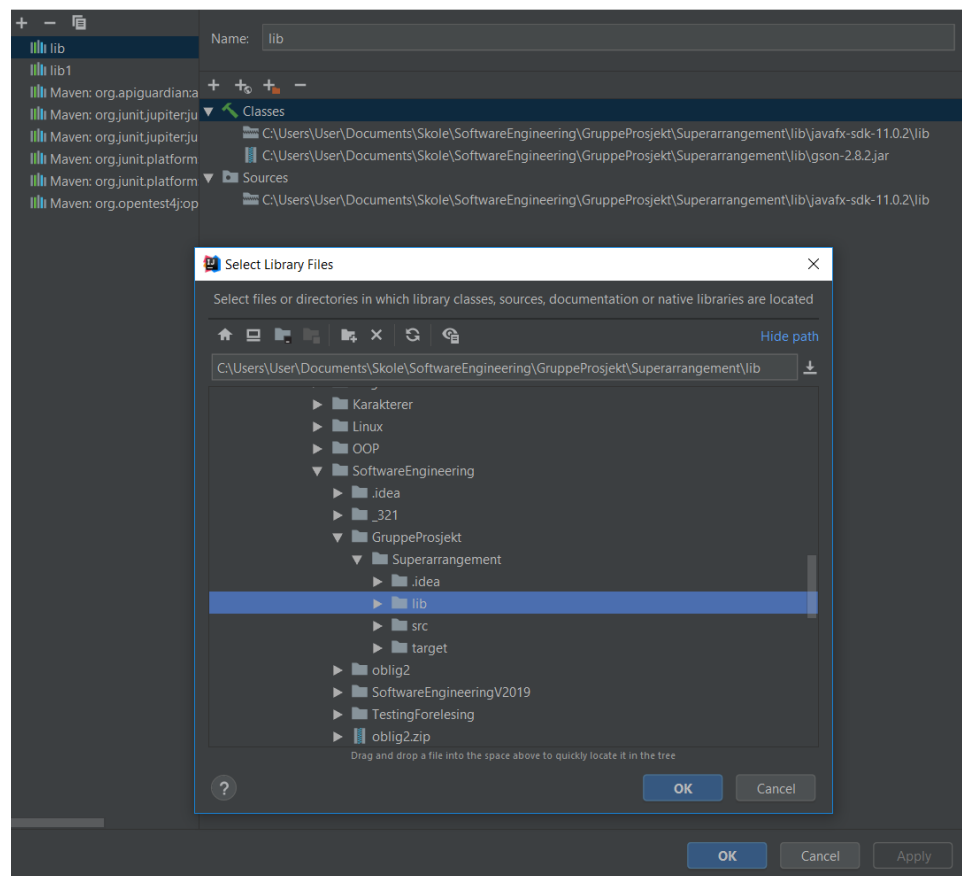
7.1. Veiledning for å legge til JavaFX og gson:

7.1.1. Legg til library

- 1. Trykk på pluss for å lage et nytt library**

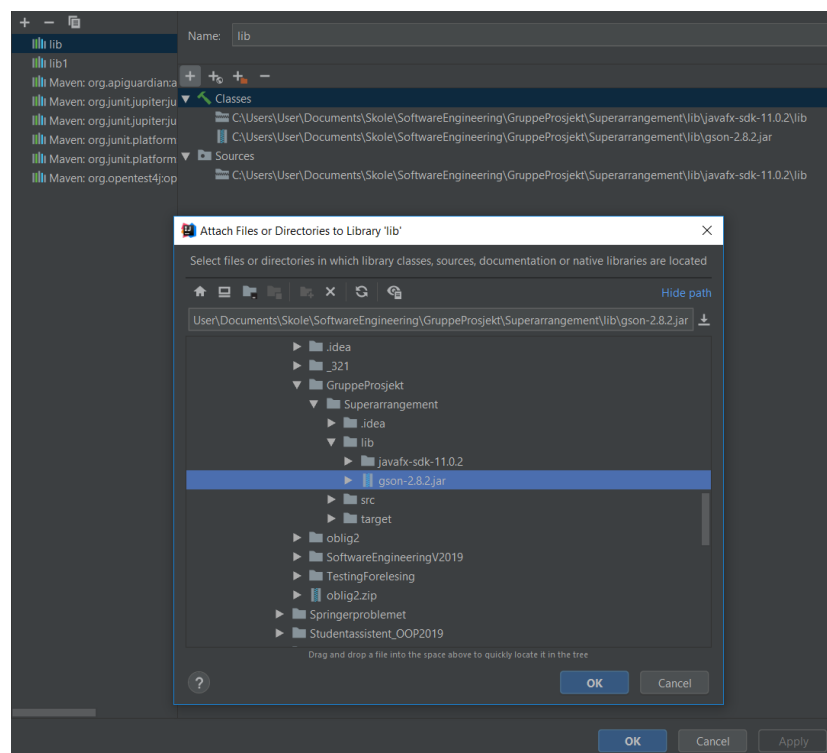


2. Velg /lib i prosjektet



7.1.2. Gson

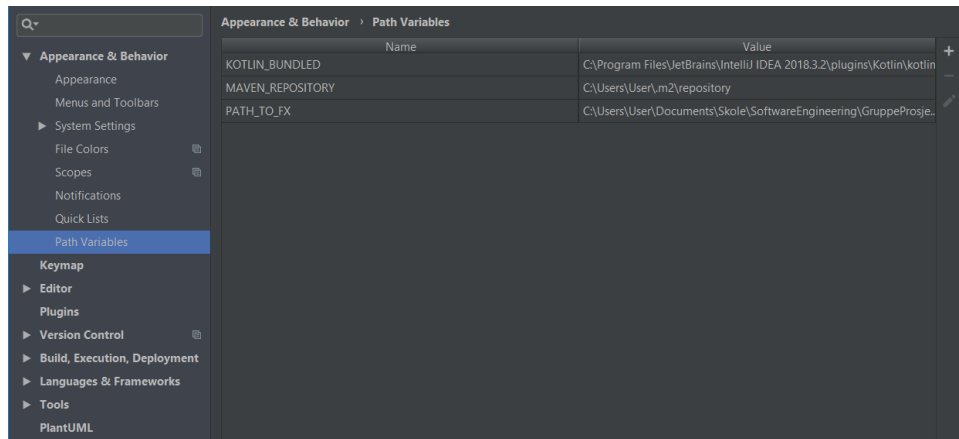
3. File -> Project structure (Ctrl-Alt-Shift-S) -> Libraries



4. Velg gson-2.8.2.jar

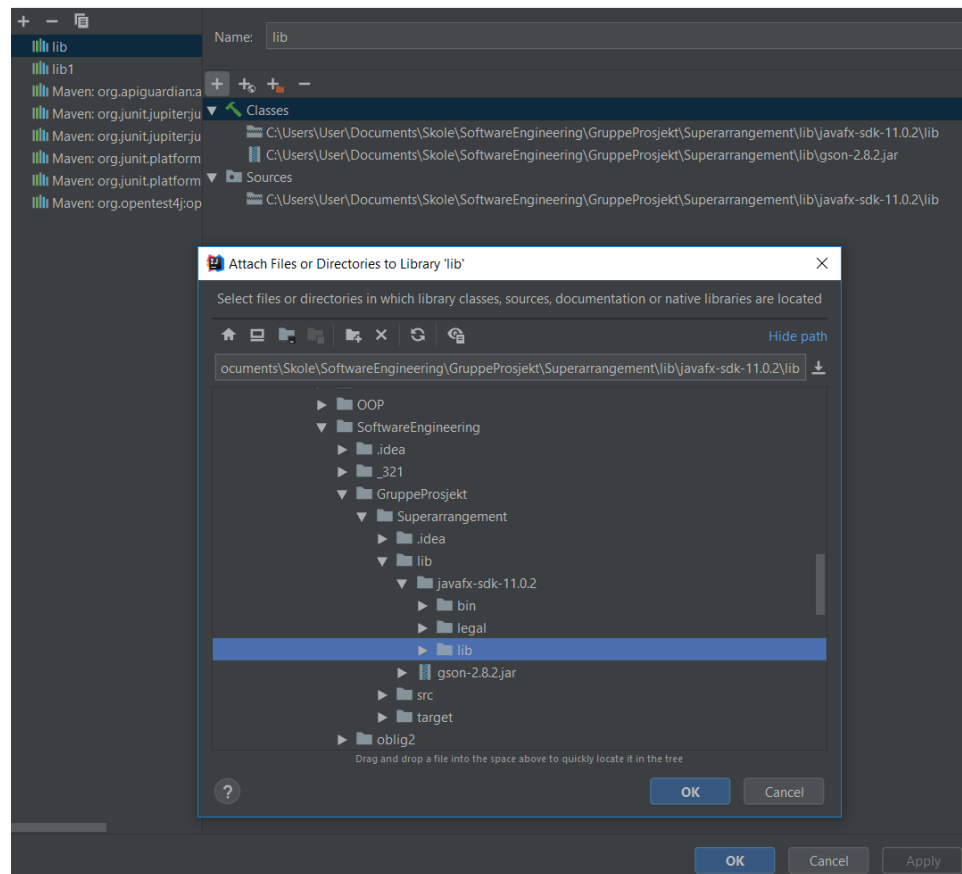
7.1.3. JavaFX.

5. Legg til en Path variable til JavaFX, **File -> Settings (Ctrl-Alt-S) -> Apperance & Behavior -> Path Variables.**



6. Trykk på pluss øverst i høyre i listen.
7. Gi variabelen navnet PATH_TO_FX og bla igjennom filer og finn
“Superarrangement\lib\javafx-sdk-11.0.2\bin”

8. Åpne project structure **File -> Project structure (Ctrl-Alt-Shift-S) -> Libraries**

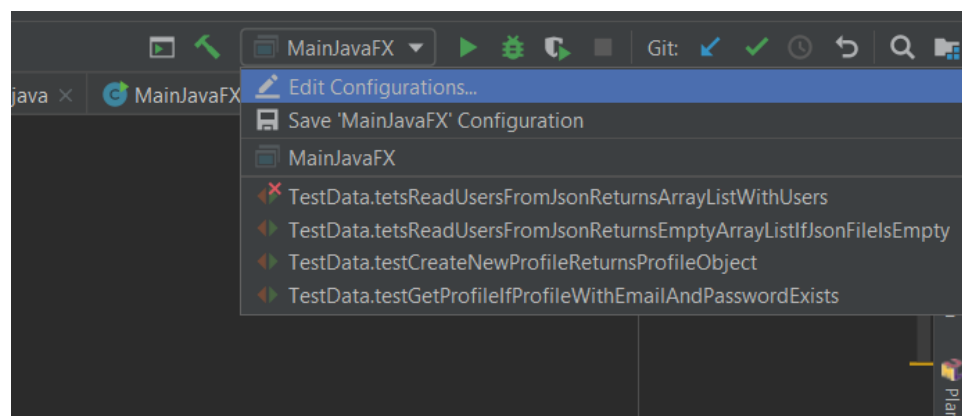


9. Trykk på pluss over listen til høyre (Valgt i bildet).

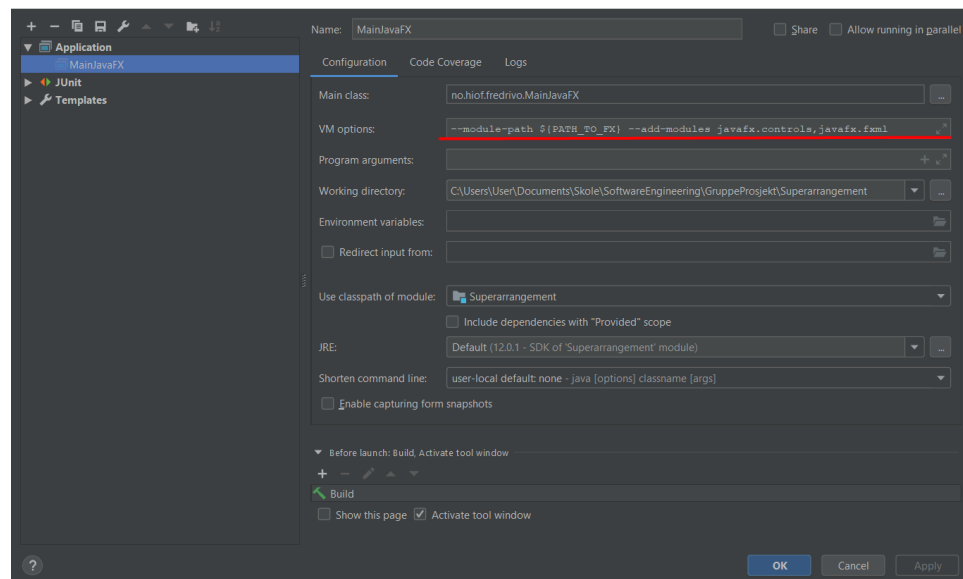
10. Legg til **“Superarrangement\lib\javafx-sdk-11.0.2\lib”**

11. Trykk ok og apply og kryss ut av Project Structure.

12. Når du skal kjøre **MainJavaFX.java** må du legge til VM options får å få kjørt programmet.



13. Velg Edit Configuration



14. Kopier teksten

--module-path \${PATH_TO_FX} --add-modules javafx.controls,javafx.fxml

og lim inn i feltet for VM options.

Prototypen er skrevet med MVC (Model, View, Controller). Her er koden delt opp i mapper for bedre struktur.

- Model:
 - Events.java
 - Profile.java
- View:
 - CartBox.fxml
 - Events.fxml
 - LoginPage.fxml
 - Profile.fxml
 - RegisterUser.fxml
- Controller:
 - SuperController.java
 - CartBoxController.java
 - EventsController.java
 - LoginController.java
 - Navigation.java

- ProfileController.java
- RegisterController.java
- Data:
 - DataHandler.java
 - Receipt.java
- Validation:
 - InputValidation

Model inneholder klasser for profil og arrangement objektene. I programmet lages det objekter som DataHandler.java serialiserer til .json filer.

View inneholder grensesnittet til visningene som består av fire hovedsider, Login, brukerregistrering, profilvisning og arrangementer, og cartBox som er vinduet der brukeren betaler for arrangementet.

Controller inneholder funksjonalitet til det som skjer i view'et. Når knapper trykkes på, tekst vises, lister oppdateres osv. Hver controller arver fra klassen SuperKontroller.java som inneholder funksjonalitet for knappetrykk som er felles for alle kontrollere. I dette tilfellet er de felles funksjonalitetene for menyknappene som tar brukeren til login-, registrering-, profil- og arrangement-sidene.

Navigation.java er en klasse som inneholder metoder som åpner sidene til menyknappene og viser AlertBokser.

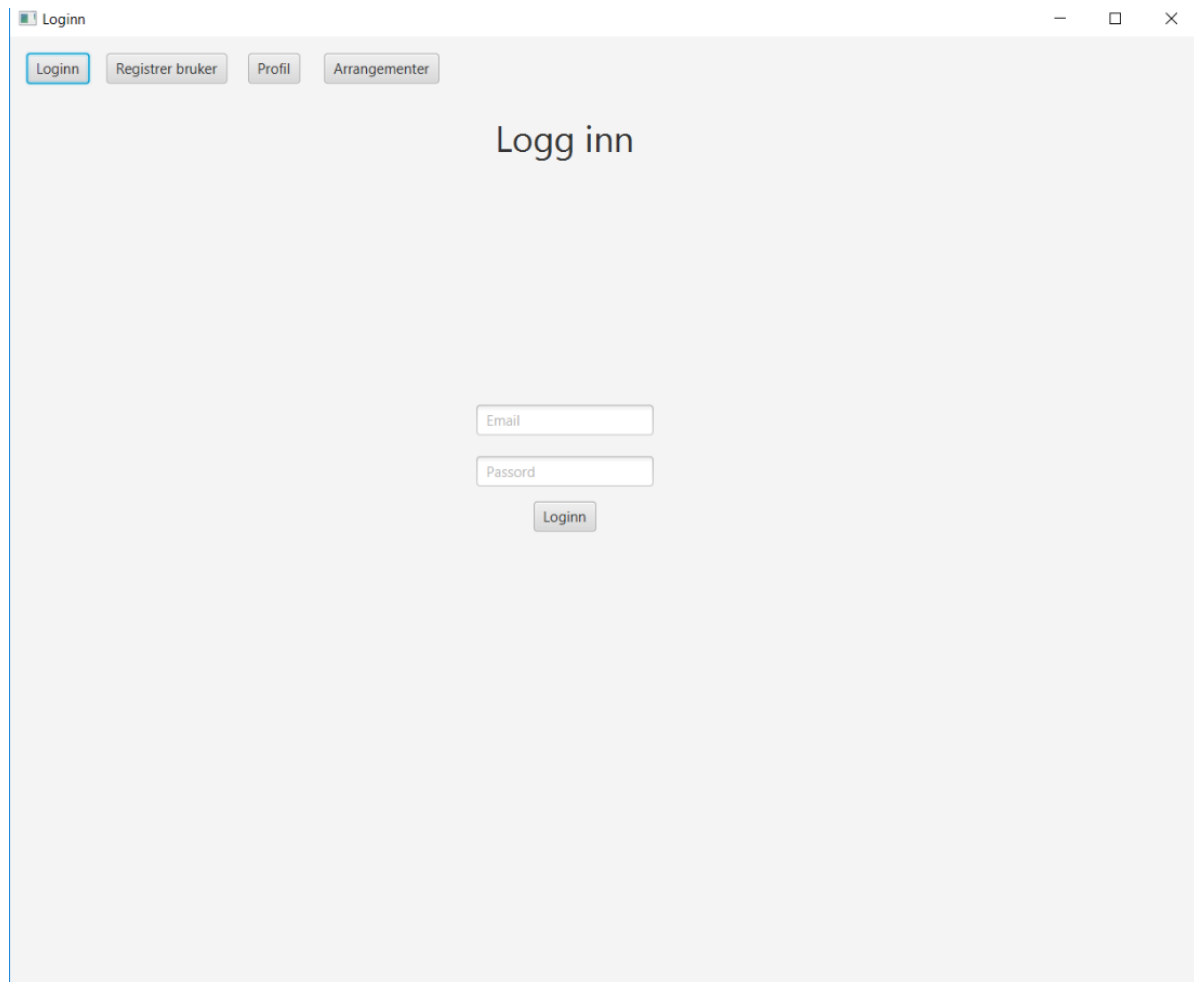
Data inneholder klassene som behandler data i programmet. DataHandler.java inneholder funksjonalitet for å lese og skrive til .json filer samt lagring av profilobjekter. Receipt.java lager kvittering etter brukeren har betalt for arrangement. Denne kvitteringen skal sendes på email men her lages det bare en tekstfil med informasjon om kjøp.

Validation inneholder funksjonalitet for inputvalidering for registrering og logg inn, og se om en bruker allerede eksisterer.

7.2. Prototypens layout og funksjon

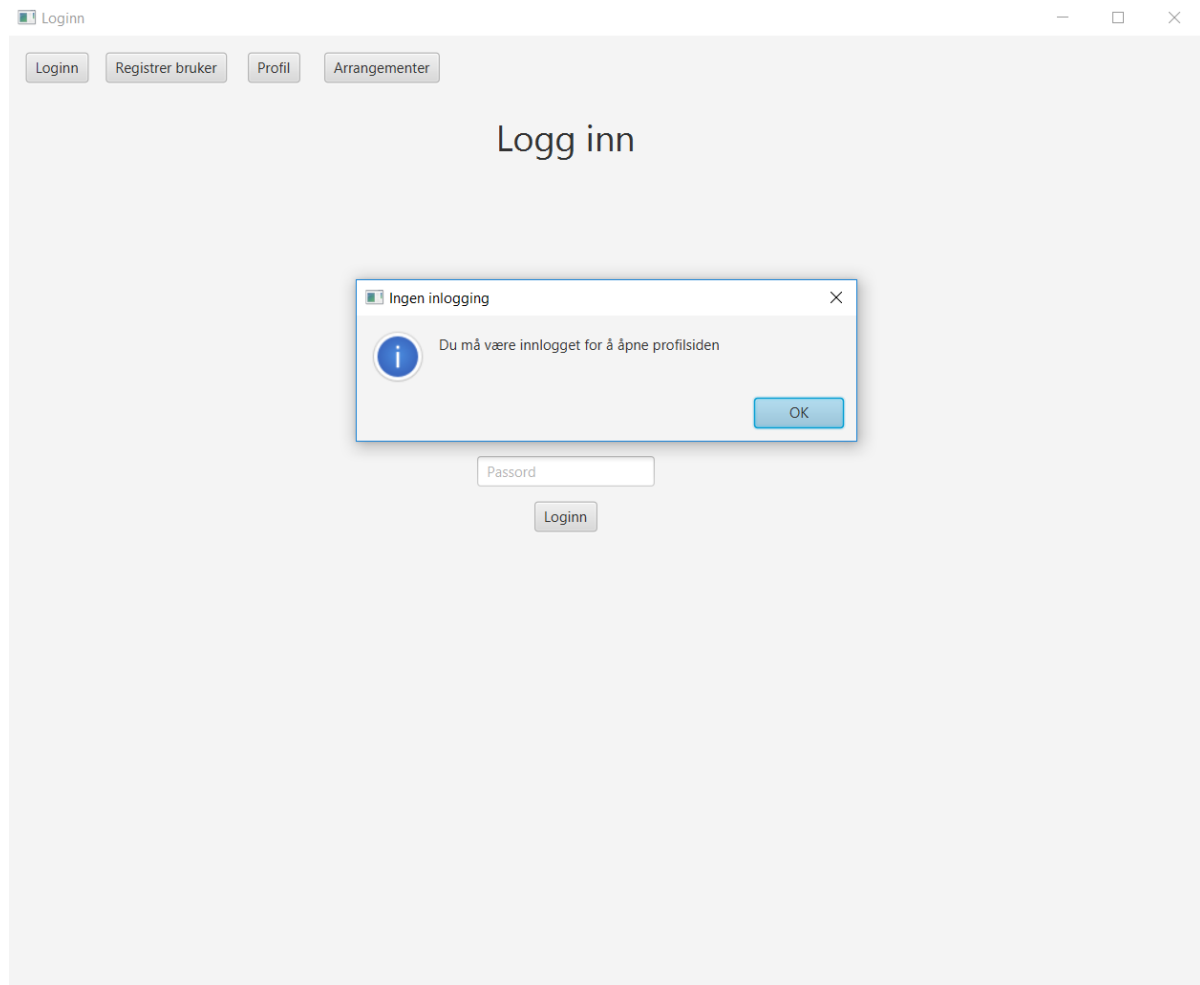
Logg inn siden er startsidene. Øverst på siden er menyknapper som tar brukeren til andre sider i prototypen.

Hvis brukeren ikke har en profil på systemet må den registrere en ny bruker.



The screenshot shows a web browser window titled "Loginn". At the top, there is a horizontal menu with four buttons: "Loginn" (highlighted with a blue border), "Registrer bruker", "Profil", and "Arrangementer". The main content area has a large heading "Logg inn" in the center. Below the heading, there are two input fields: "Email" and "Passord". Below these fields is a "Loginn" button.

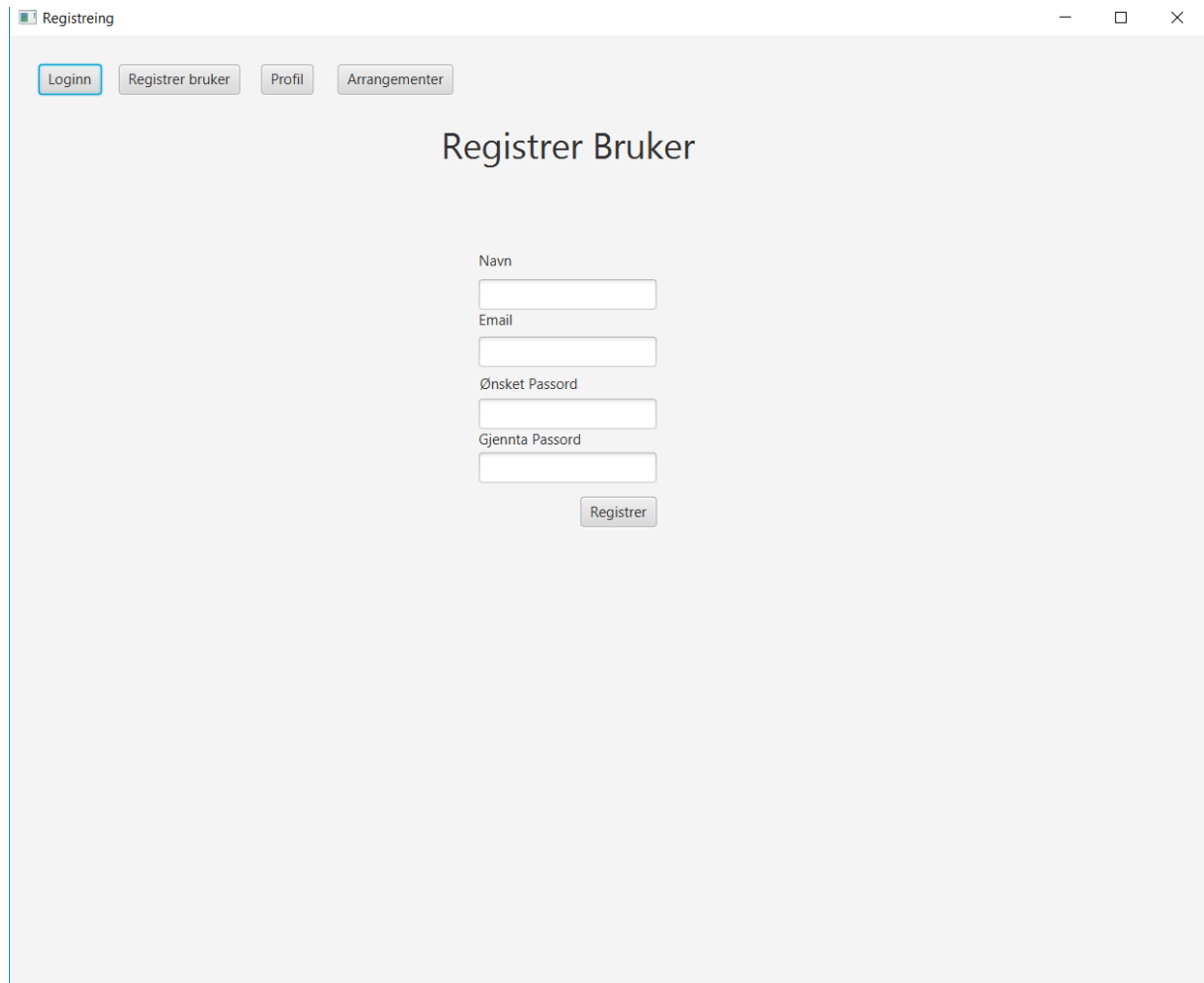
FIGUR 12: PROTOTYPE LOGG INN



FIGUR 13: PROTOTYPE PROFILSIDEVISNING FEILMELDING

Kravene for registrering er følgende:

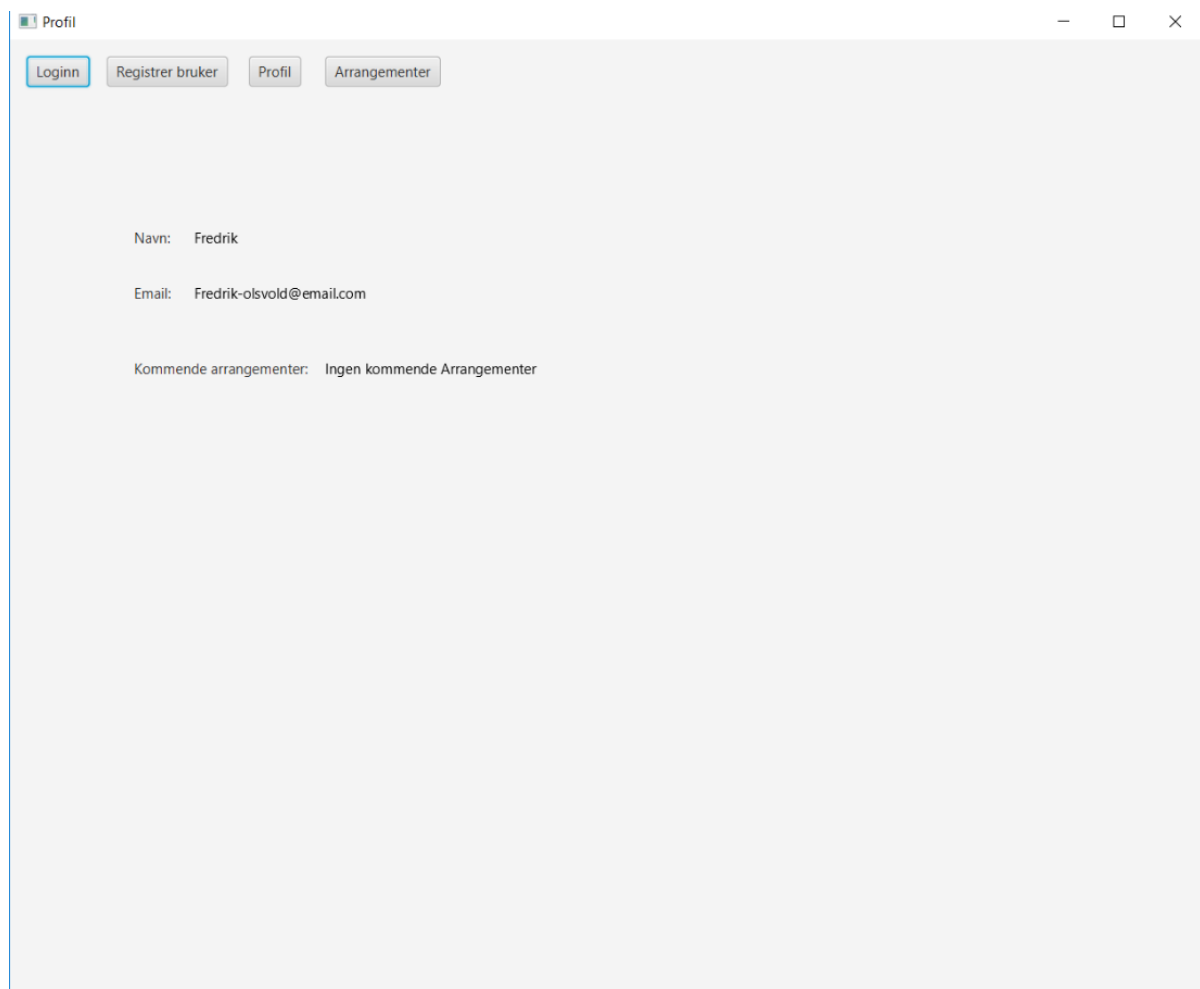
1. Navn kan bare inneholde store og små bokstaver og mellomrom.
2. Email kan bare inneholde store og små bokstaver, tall og tegnene _!#\$%&'*/+=?`{|}~^.- er lovlig før @. Alfakrøll er forventet og domenet på slutten må inneholde minst to tegn og maks fire.
3. Passord må inneholde minst 8 tegn, minst en liten bokstav, en stor bokstav og minst ett tall.



The screenshot shows a web browser window with the title 'Registreing'. At the top, there are four buttons: 'Loginn', 'Registrer bruker', 'Profil', and 'Arrangementer'. The 'Loginn' button is highlighted with a blue border. Below these buttons, the heading 'Registrer Bruker' is centered. Under the heading, there are four input fields labeled 'Navn', 'Email', 'Ønsket Passord', and 'Gjennta Passord'. Below the 'Gjennta Passord' field is a 'Registrer' button.

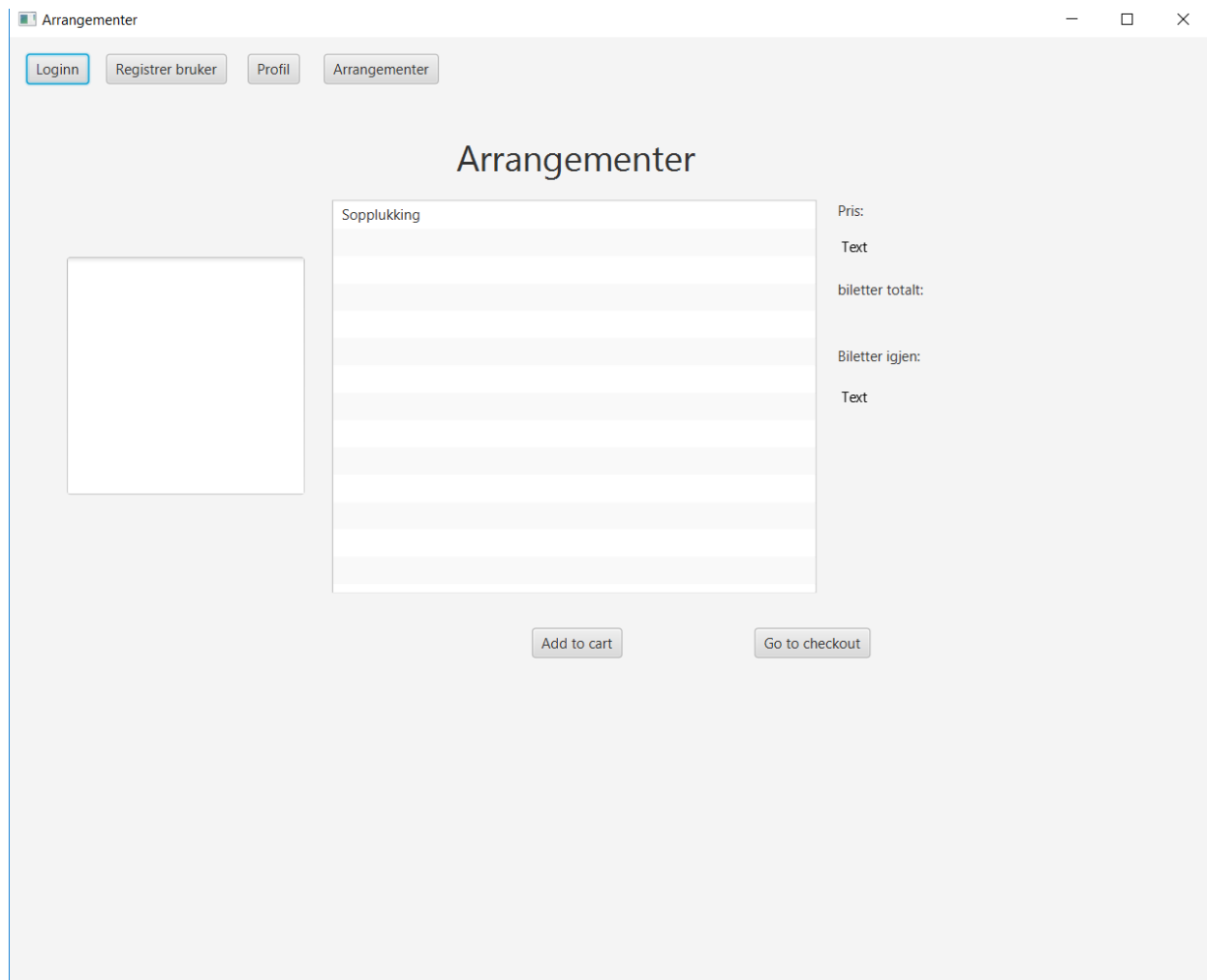
FIGUR 14: PROTOTYPE REGISTRERE BRUKER

Når en bruker er registrert blir den sendt til profilsiden som viser oversikt over profilen.

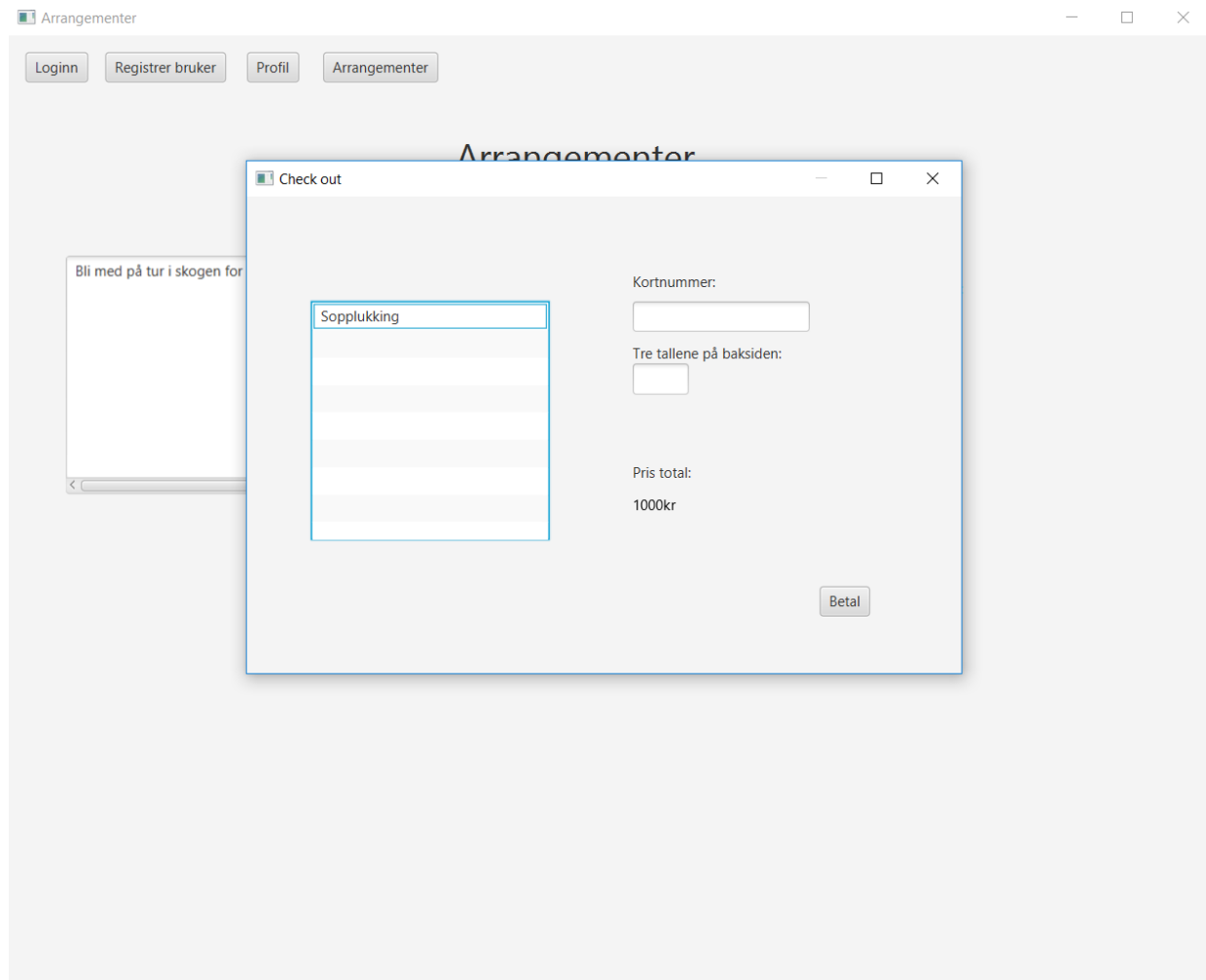


FIGUR 15: PROTOTYPE PROFILSIDE

Den nye brukeren er innlogget og kan nå kjøpe billetter til arrangementer på arrangement-siden. Brukeren velger et arrangement fra listen og legger det til i varekurven. Etter lagt til arrangementer kan brukeren gå til kassen og betale for billettene.

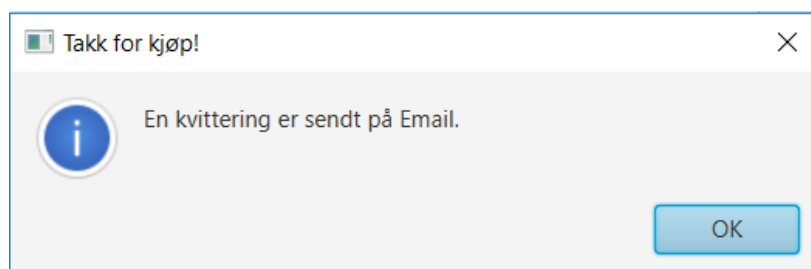


FIGUR 16: ARRANGEMENT VISNING



FIGUR 17: BETALING MED KORT

Brukeren betaler for billetten og får kvittering sendt på email. Prototypeversjonen av systemet lager bare kvitteringene og lagrer de lokalt inntil videre, men har mulighet for videreutvikling.



FIGUR 18: PROTOTYPE KVITTERING

7.2.1. Stubs.

Funksjoner som simulerer hvordan systemet skal utføre visse ting er:

- Lagring av brukere og arrangementer
- Kvitteringer
- Valg av billetter til arrangementer

- Kjøp av billetter.

Lagring av brukere og arrangementer skal i fremtidig versjon av systemet bruke Database for lagring, men vi bruker json for proof-of-consept.

Kvitteringer lages som en tekstfil og lagres lokalt, men i realitet skal denne sendes som e-post. Noe som skal videre utvikles for fullstendig system.

Valg av billetter skal vises som bilder med overskrift. Listevisningen er en veldig forenklet versjon av dette.

Kjøp av billetter skal kunne betales på flere måter enn bare bankkort. I prototypen er det bare mulig for kortbetaling men det er mulighet for videreutvikling.

8. Testing

Under utviklingen av systemet som er produsert, og i forbindelse med kravoppfylling og sikring av funksjonalitet, må det gjøres tester som sørger for at hvert steg i utviklingsprosessen er utfylt og tatt hensyn til.

Det er flere tester som må gjøres på et system før det kan regnes som komplett. Enhetstesting gjøres av utviklere, og det underveis i programmering fasen. Der enkelte deler av applikasjonen blir testet for seg selv. Integrasjonstest inngår flere komponenter, der det testes om de kommuniserer som det skal sammen og gjør jobben den er programmert til. Systemtesting tester systemet som helhet, og kjøres når det er fått til større deler av systemet. Til slutt har vi akseptansetest som utføres av sluttbrukeren, der systemets komponenter sjekkes mot kravene som ble satt.

Når det gjelder servernes kapasitet så må de stress-testes. Det gjøres for å sikre at serveren takler alle forespørsler som vil komme, og det samme må databasen. Et sikkert system vil kunne takle de mange forespørslene uten å få noe nedgang.

8.1. Tester som ikke er Junit tester i prototypen

8.1.1. Registrering av bruker

Tittel:

Registrering av nye brukere

Beskrivelse:

En bruker skal kunne registrere ny bruker på Superarrangement.no

Antagelser:

Det blir brukt en støttet nettleser til gjennomføring

Test steg:

1. Gå til Superarrangement.no
2. Trykke på "Registrering" knappen
3. I feltet "Navn" skal fornavn og etternavn fylles
4. I feltet "E-post" skal gyldig e-post adresse fylles inn
5. Bruker fyller ønsket passord i feltet "Passord" etter kravene
6. I feltet "Bekreft passord" skal passordet skrives inn på nytt
7. Bruker skal trykke "Register bruker"

Forventet resultat:

Innlogget side skal dukke opp, hvor bruker får sett inn i egen profil og videre til arrangementer

8.1.2. Fakturering av bruker

Tittel:

Fakturering av bestillinger - betaling

Beskrivelse:

Teste om serverens betalingssystem fungerer som den skal, og fakturerer riktig bruker med riktig beløp

Antagelser:

Brukeren har valgt arrangement, og tastet inn riktig kortinfo i betalingsskjema

Test steg:

1. Bruker velger ønsket arrangement
2. Bruker trykker "Add to cart" knappen
3. Bruker trykker på "Go to checkout"
4. Bruker fyller inn kortinformasjon
5. Bruker trykker deretter "Betal" knappen

Forventet resultat:

En beskjed som forteller at "Kvittering er sendt på e-post" og en bekreftelse blir sendt på e-post til bruker

8.1.3. Bestilling av arrangement

Tittel:

Bestilling av arrangement - booking

Beskrivelse:

Teste om systemet legger til ønsket bestilling i handlekurven til bruker, deretter vise oppdatert liste med bestilte arrangementer

Antagelser:

Brukeren har logget inn på profilen sin, og valgt "Arrangementer"

Test steg:

1. Bruker trykker på "Arrangementer" knappen
2. Bruker velger
3. Bruker trykker på "Go to checkout"
4. Bruker fyller inn kortinformasjon
5. Bruker trykker deretter "Betal" knappen

Forventet resultat:

En beskjed som forteller at "Kvittering er sendt på e-post" og en bekreftelse blir sendt på e-post til bruker.

8.2. Tester som er Junit tester i prototypen

8.2.1 Input tester

Tittel:

Logg inn felter er tomme

Beskrivelse:

Systemet skal si ifra til brukeren hvis ingen av feltene under logg inn er fylt ut ved logg inn knappetrykk

Antagelser:

Brukeren har ikke skrevet noe i tekstfeltene Email eller Passord.

Test steg:

1. Bruker trykker på logg inn knappen.

Forventet resultat:

En beskjed som forteller at brukeren må fylle i begge tekstfeltene før den kan logge inn.

Junit test:

```
@Test
public void testLoginInputEmptyCheckIsEmpty() {
    assertTrue(InputValidation.loginInputEmptyCheck( email: "", password: "" ));
}
```

8.2.2. Data tester

Tittel:

Bruker eksisterer allerede

Beskrivelse:

Systemet skal si ifra til brukeren hvis den registrerer seg med en eksisterende e-post i systemet.

Antagelser:

E-post er lagret i systemet.

Test steg:

1. Bruker skriver inn e-post under registrering.
2. Bruker trykker på registrer bruker.

Forventet resultat:

En beskjed som forteller at brukeren allerede eksisterer i systemet.

JUnit test:

```
@Test
void testGetProfileIfProfileWithEmailAndPasswordExists() {
    //Metoden henter en Profil fra Json filen users.json
    //En profil med emailen testemail@email.com og passordet finnes Apekatt123
    assertNotNull(DataHandler.getProfile(email: "testemail@email.com", password: "Apekatt123"));
}
```