

Billett- og arrangement applikasjon



Laget av gruppe 14

Enis Jasharaj - Enisj@hiof.no - Studentnr: 173166

William Svea-Lochert - williams@hiof.no - Studentnr: 173198

Martin Øistad Skåksrud - Martinos@hiof.no - Studentnr: 173185

Ingrid Elise Krogh Dahl - iedahl@hiof.no - Studentnr: 173017

Elin Skaret Larald - elin.s.larald@hiof.no - Studentnr: 173022

Dato: 26.04.2019

Kurs: Software Engineering and testing

1. Bakgrunn og problemstilling for prosjektet	4
2. Domene beskrivelse	4
3. Løsning → hvordan håndterer løsningen de forskjellige elementene i problemstillingen?	5
4. Personas	6
5. Use case	9
5.1 Brukssituasjoner for arrangør	9
5.2 Brukssituasjoner for kunde	13
5.3 Brukssituasjoner for kinosjef	17
6. Krav	21
6.1 Funksjonelle krav	21
Kinosjef/administrator av systemet	21
Registrering og innlogging av arrangør	22
Oppretting av arrangement	22
Administrasjon av eksisterende arrangement	22
Registrering og innlogging av kunde	23
Søke opp og filtrere arrangement	23
Bruk av kundens posisjon	23
Booking - valg av billett og sete	23
Betalning og verifisering	23
Adgangskontroll	24
6.2 Ikke-funksjonelle krav	24
Kunde	24
Arrangør	25
Arrangement og betaling	25
Databehandling og plattformer	25
6.3 Estimering av krav til hovedsystemet	26
6.3.1 - Funksjonelle krav	26
6.3.2 - Ikke funksjonelle krav	26
7. Prototypen	27
7.1 Krav for å kjøre prototypen	27
7.2 Begrensninger og antagelser for prototypen	27
7.3 Oversiktstabell over krav og viktighet for prototypen	29
7.4 Forklaring til krav og deres viktighet	29
7.5 Funksjonelle krav implementert i prototypen	30
7.5.1 Registrering og innlogging av arrangør.	30
7.5.2 Oppretting av arrangement	31
7.5.3 Registrering og innlogging av kunde	31
7.5.4 Søke opp og filtrere arrangement	32
7.5.5 Booking - valg av billett og sete	32

7.5.6 Betaling og verifisering	32
7.5.7 Adgangskontroll	32
7.6 Ikke funksjonelle krav implementert i prototypen	33
7.6.1 Arrangement og betaling	33
8. Sekvensdiagram	34
8.1 Oppretting av arrangement	34
8.2 Innlogging for kunder	35
8.3 Innlogging for arrangører	36
8.4 Søk etter kategori	37
8.5 Søk etter arrangement	38
8.6 Opprette kundekonto	39
8.7 Opprette arrangør konto	40
8.8 Booke billetter og betaling	41
9. Aktivitetsdiagram	42
9.1 Registrere kunde- og arrangør konto	42
9.2 Innlogging for arrangører og kunder	43
9.3 Bestilling og betaling av billetter	44
9.4 Søk av arrangement og kategori	45
9.5 Opprette arrangement	46
10. Beskrivelse av prototypen	47
10.1 Prototypen og det endelige systemet	48
10.2 Skjermbilder fra prototype	49
Forside	49
Opprette arrangør konto	49
Arrangør seksjon	50
Opprette arrangement	50
Søk etter arrangement og kategori	50
Kjøp av billetter	51
Se mine billetter	52
11. Kjente svakheter og problemer i prototypen	52

1. Bakgrunn og problemstilling for prosjektet

Prosjektet har sin bakgrunn i at underholdnings bildet i norske byer stadig blir større, både i antall arrangement og bredden på dette tilbudet. I denne sammenhengen ønsker man et system for at enkeltpersoner eller bedrifter skal kunne opprette egne arrangement. Systemet skal styre salg av billetter, lage unike billetter, ta imot innbetalinger og håndtere adgangskontroll til arrangementer.

Etter forespørsel fra de lokale kinoene i regionen skal dette prosjektet levere et system som skal kunne brukes universelt - både hos de respektive kinoene og som støtte hos forskjellige andre aktører. Andre aktører kan for eksempel være lokale idrettslag, mindre konserter osv. Med universelt menes det at systemet skal være et billettsystemet for de lokale kinoene, men at også enkeltpersoner/bedrifter skal ha mulighet til å bruke det som billettsystem til egne arrangementer.

2. Domene beskrivelse

Overordnet er en billettapplikasjon en måte å gjennomføre kjøp og salg av billetter til diverse arrangementer lettere for både arrangør og kunde. Istedenfor at kjøp og salg foregår i såkalte "billettluke" - hvor man må stå i kø for å vente på sin tur. Kan man via en applikasjon gjennomføre hele handelen på en datamaskin, smarttelefon, nettbrett og lignende.

En billettapplikasjon innehar funksjonalitet som lar vanlige personer, bedrifter eller organisasjoner fungere som arrangører for å publisere sine arrangementer i applikasjonen. Arrangementene skal typisk kunne opprettes med tilhørende informasjon som for eksempel pris, antall tilgjengelige billetter, aldersgrense og lignende. For at applikasjonen skal skape verdi for arrangøren - må den også legge til rette for kjøp og salg - altså inneha en betalingsløsning.

I tillegg til det som er nevnt over, inneholder også billettapplikasjoner en form for adgangskontroll for å sørge for at ingen billetter kan bli gjenbrukt.

I likhet med salg i billettluke - hvor man alltid har kontroll på hvor mange billetter som er igjen etter fortløpende salg skal også billettapplikasjon ha kontroll på.

Fra kundens perspektiv skal billettapplikasjonen sørge for at det er lett å finne fram til sitt ønsket arrangement - og gjennomføre betalingen. For at applikasjonen skal støtte dette, lar den typisk kunden søke etter arrangement de kan tenke seg, eller filtrere basert på egne preferanser.

3. Løsning → hvordan håndterer løsningen de forskjellige elementene i problemstillingen?

Gruppens løsning vil være å lage en Webapplikasjon som lar de lokale kinoene drive sin drift som vanlig med tanke på registrering av kinofilmer, booking av kinobilletter, betalingsløsning og adgangskontroll, samtidig som det skal være mulig for andre aktører å gjennomføre samme funksjonalitet rettet mot egne arrangementer.


Hvordan systemet løser elementene i problemstillingen:

- *Registrering av arrangement*
 - *For at en arrangør skal kunne registrere arrangementer, må systemet utvikles slik at en bruker selv skal kunne definere navn, sted, antall tilgjengelige billetter, aldersgrense osv selv i systemet.*
- *Betaling*
 - *Kunden skal kunne velge mellom to forskjellige betalingsløsninger, som er kortbetaling med Visa/Mastercard og Vipps. Denne funksjonaliteten skal ikke utvikles fra bunnen av, men det skal hentes fra de eksterne leverandørene Stripe og Vipps.*
- *Booking*
 - *For at en kunde skal kunne booke en eller flere billetter, må systemet ha en løsning for å kunne søke etter og filtrere arrangement. Dette må ligge til rette slik at den enkelte kunde får booket arrangement etter eget ønske. Når kunden har valgt billett og fått betalingen godkjent, så skal billetten(e) bli sendt på mail med en bekreftelsesmelding.*


En mer beskrivende løsning av hvordan problemet er løst finnes lenger ned i dokumentasjonen. Det ligger vedlagt en beskrivende kravspesifikasjon, modeller med tilhørende forklaringer og skjermbilder som viser flyten i prototypen.

4. Personas

I denne delen blir det presentert tre ulike persons segmenter som er interessenter av applikasjonen. Disse er fiktive personer, hvor hver av dem beskriver sine ønsker, problemer og mål med med billettapplikasjonen.

Navn og alder	Detaljer	Mål
 Navn: Johnny Bravo Alder: 51	<p>Rolle: arrangør</p> <p>Johnny er fra- og bor i Norges desidert fineste by, Fredrikstad. Johnny jobber til vanlig som markedsansvarlig i et konsulentfirma med 35 ansatte med hovedkvarter i hjembyen. Hverdagen til Johnny er temmelig fullpakket med jobb og familieliv – noe som gjør hans tilleggjobb/hobby en smule vanskelig – nemlig hovedansvarlig for byens lokale underholdningsavdeling som stiller i stand noen arrangementer av forskjellig type hvert år. Disse kan variere fra fotball fester, barnas aktivitetsdag, Fredrikstad BBQ festival osv. Med jobb, familie og som arrangementsansvarlig blir hverdagen til Johnny preget av mye stress.</p> <p>Johnny ønsker da et intuitivt system som lar han registrere aktuelle arrangementer, med respektiv info som billett antall, dato, sted, sjanger samt at systemet skal legge til rette for kjøp, salg og adgangskontroll. Per dags dato er dette ting som tar mye av Johnnys tid på ettermiddagene, ettersom billetter, kjøp/salg, adgangskontroll må kombineres med forskjellige aktører og tjenester.</p>	<p>«Jeg har lyst til å ha mer tid til overs jeg kan bruke på de virkelige viktige tingene i livet mitt. Jeg er veldig glad i de to jobbene mine, men skulle så gjerne hatt mer tid til familien min!»</p>

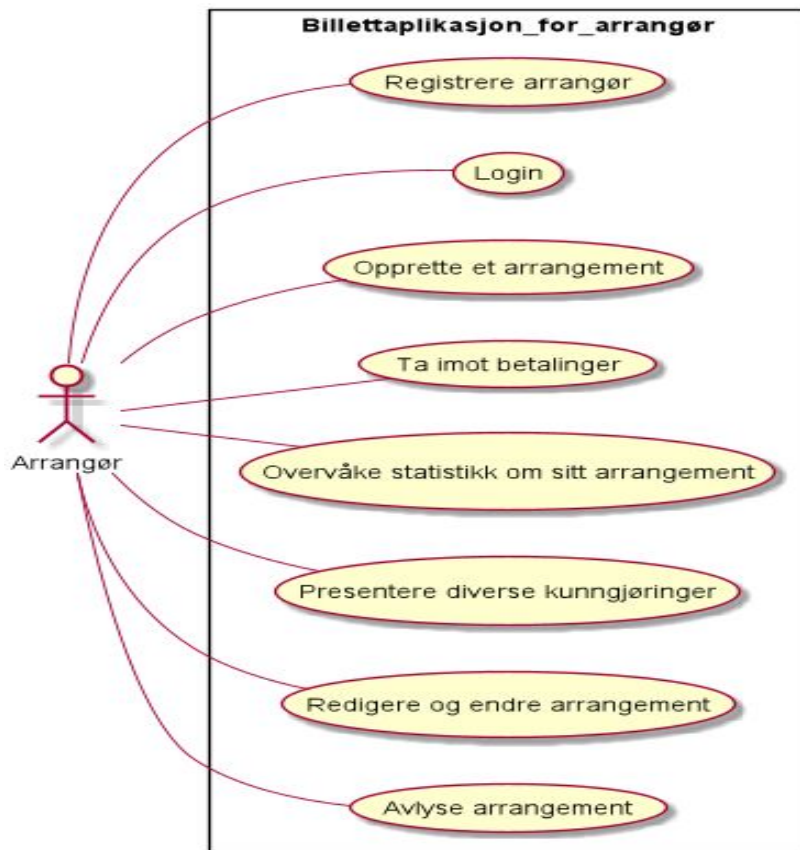
Navn og alder	Detaljer	Mål
 <p>Navn: Scooby Doo Alder: 20</p>	<p>Rolle: kunde/bruker</p> <p>Scooby er opprinnelig fra Sokna, men har nylig flyttet til Fredrikstad for å studere Ingeniørfag. Han bor i et kollektiv sammen med 5 andre studenter som studerer det samme som han. I likhet med de fleste andre studenter – tar Scooby og vennene hans seg regelmessig tid til å ha det litt moro for å tenke på noe annet enn bare skole. Da de studerer i den sprudlende byen, Fredrikstad - er det stadig nye arrangementer som avholdes. Per dags dato må Scooby og vennene ta bussen til den lokale kinoen som støttes av en distributør av billetter til forskjellige arrangementer. De aller fleste i kollektivet inkludert de andre kollektivenes medlemmer er enige om at de syns det er tungvint og måtte reise et lite stykke – kun for å få tak i en billett til arrangementer.</p> <p>Scooby ønsker at det skulle finnes et system som gjør det slik at han slipper å reise til den lokale kinoen hver gang han vil kjøpe en billett. Han syns det hadde vært supert hvis systemet lot han søke opp arrangementer han er interessert i, eller la han søke etter arrangementer basert på forskjellige typer/sjangere av arrangementer. I tillegg vil han kunne betale for billetten via kort eller mobil med en gang han velger den – slik at det eneste som står igjen er å glede seg til å ha det moro.</p>	<p>«Jeg vil kunne ligge behagelig på sofaen å søke etter og kjøpe billetter til de arrangementene jeg ønsker – UTEN å måtte reise med bussen inn til sentrum!»</p>

Navn & alder	Detaljer	Mål
 <p>Navn: Kinosjef Rino Alder: 67</p>	<p>Rolle: Kinosjef</p> <p>Kinosjef Rino er født og oppvokst i verdens fineste by - Fredrikstad. Rino er en teknologi interessert person som alltid er på utkikk etter det beste utstyret og de beste systemene i sin kino.</p> <p>Ved siden av jobben som kinosjef, er Rino samtidig brennende opptatt av byens andre kulturelle tilbud. I og med at det allerede er mulig for mennesker å dra til kinoen for å kjøpe billetter til andre arrangementer enn kinoens spillefilmer - ser Rino at det er forbedringspotensiale med tanke på hvordan disse menneskene lettere skal kunne få tak i billetter til ønskede arrangementer.</p> <p>Rino ønsker da å få utviklet et system som skal kunne fungere universelt. Han vil at kinoen hans skal kunne bruke systemet til å selge kinobilletter, samtidig som andre mindre aktører skal kunne registrere sine arrangementer for kjøp og salg via systemet.</p>	<p>“Jeg vil at det skal være mulig for de som ønsker å arrangere noe, skal få mulighet til å drive kjøp og salg via et intuitivt system”</p>

5. Use case

Diagrammene under viser til de forskjellige brukssituasjonene de forskjellige aktørene og systemet samhandler slik at det gir verdi for aktøren. I tillegg til selve bildet, ligger det også en dypere forklaring til hver brukssituasjon under hvert diagram.

5.1 Brukssituasjoner for arrangør



Use Case	Registrere arrangør
Aktør	Arrangør
Forklaring	Å registrere seg som en arrangør i systemet, vil gjøre at du har muligheten til å bruke systemet for å håndtere billettsalg og adgangskontroll til dine arrangementer. Man velger "registrer ny arrangør" i menyen og registrerer seg med firmanavn, fornavn, etternavn, e-post og passord. Etter å ha opprettet konto kan man logge inn med denne.
Stimulus	Arrangør ønsker å opprette en konto i systemet
Respons	Arrangør konto er opprettet

Use Case	Login
Aktør	Arrangør
Forklaring	For at man skal kunne bruke systemet som en arrangør er det påkrevd at man logger inn for å verifisere at man faktisk er en arrangør. Systemet ber deg om å logge inn med din e-postadresse samt passord.
Stimulus	Arrangør ønsker å logge inn i systemet og taster inn login informasjon
Respons	Arrangør er logget inn og kan ta i bruk systemet.

Use Case	Opprette arrangement
Aktør	Arrangør
Forklaring	For at en arrangør skal kunne utnytte systemet så godt som mulig, er det essensielt at en arrangør skal kunne opprette ønskede arrangementer. En arrangør oppretter et arrangement med følgende informasjon: arrangementets navn, kategori, beskrivelse, dato, sted, aldersgrense og antall billetter.
Stimulus	Arrangør registrerer arrangement med tilhørende informasjon.
Respons	Arrangementet er opprettet; arrangement lagt til i liste over arrangementer

Use Case	Ta imot betalinger
Aktør	Arrangør
Forklaring	For at en arrangør skal kunne motta penger for de solgte billettene sine, skal systemet legge til rette for overføring av penger. Denne funksjonaliteten skal ikke utvikles fra bunnen av, men det skal hentes fra de eksterne leverandørene Stripe og Vipps.
Stimulus	En kunde kjøper en eller flere billetter til et arrangement.
Respons	Pengene er overført til arrangørens konto

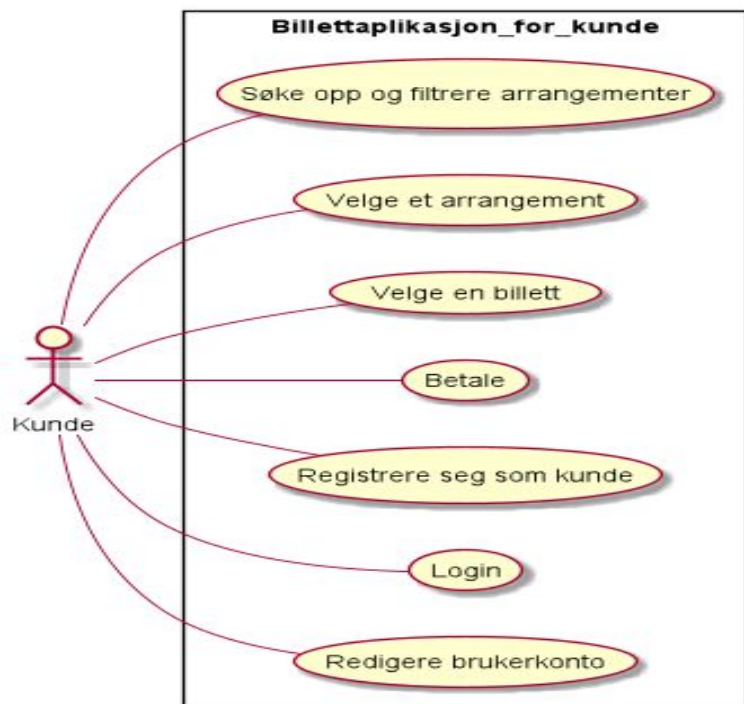
Use Case	Overvåke statistikk om sitt arrangement
Aktør	Arrangør
Forklaring	En arrangør skal til enhver tid kunne se statistikk over blant annet hvor mange billetter som er solgt.
Stimulus	En arrangør ønsker å se en oversikt over statistikken
Respons	Arrangøren blir presentert en oversikt over arrangementets statistikk

Use Case	Presentere diverse kunngjøringer
Aktør	Arrangør
Forklaring	Å la arrangøren dele diverse kunngjøringer vil skape større verdi for både arrangører og for kunder. Her vil arrangøren da ha mulighet til å for eksempel reklamere for eventuelle tilbud ved arrangementet, som for eksempel mat og drikke.
Stimulus	Brukeren skriver en kunngjøring vedrørende arrangementet
Respons	Kunngjøringen blir presentert for kunder.

Use Case	Redigere og endre arrangement
Aktør	Arrangør
Forklaring	Et eksempel på dette kan være at en arrangør vil trykke opp et sett til med fler tilgjengelige billetter. En annen ting kan være hvis det blir nødvendig å endre dato på arrangementet.
Stimulus	Brukeren registrerer ønsket endring for arrangementet
Respons	Arrangementet blir endret umiddelbart.

Use Case	Avlyse arrangement
Aktør	Arrangør
Forklaring	Det kan hende at et arrangement plutselig må avlyses grunnet sykdom - eller andre tilfeller som er mer akutte.
Stimulus	Arrangør velger å avlyse arrangement
Respons	Arrangementet er avlyst; arrangør refunderer billett kostnad

5.2 Brukssituasjoner for kunde



Use Case	Registrere seg som kunde
Aktør	Kunde
Forklaring	For at kunder kan kjøpe billetter må de registrere seg som kunde og dermed opprette en brukerkonto. Dette gjør de ved å velge "registrer kunde konto" i menyen og registrere seg med fornavn, etternavn, e-post og passord. Hvis brukerens input er ok blir kontoen opprettet og man kan bruke e-post og passord til å logge inn med.
Stimulus	Ny kunde ønsker å kjøpe billetter til et arrangement og må først opprette en brukerkonto.
Respons	Brukerkonto er opprettet og kunden kan videre logge inn i systemet for å kjøpe billetter.

Use Case	Login
Aktør	Kunde
Forklaring	For at kunder kan kjøpe billetter må de først logge seg inn med sin brukerkonto.
Stimulus	Kunde ønsker å kjøpe billetter til et arrangement og taster inn brukernavn og passord.
Respons	Kunde er logget inn og kan ta i bruk bestillingssystemet.

Use Case	Redigere brukerkonto (innbefatter også sletting av brukerkontoen).
Aktør	Kunde
Forklaring	Kunder skal selv ha mulighet til å endre alt av registrert informasjon på sin brukerkonto. Kunden skal også kunne slette brukerkontoen hvis han ikke ønsker å være kunde lenger.
Stimulus	Kunder velger funksjonaliteten for endringer eller sletting av kundekontoen og utfører endringer eller sletter kontoen.
Respons	Endringer blir oppdatert umiddelbart i systemet eller kundekontoen forsvinner umiddelbart fra systemet ved sletting.

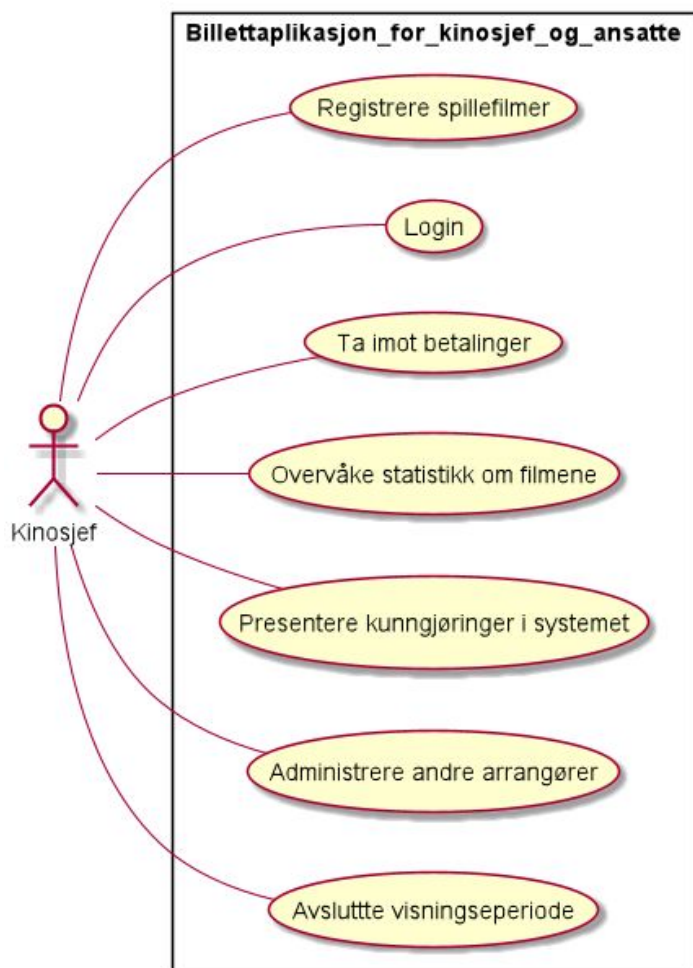
Use Case	Velge et arrangement
Aktør	Kunde
Forklaring	Etter en kunde har funnet sitt ønskede arrangement, kan han velge dette arrangementet.
Stimulus	Kunden velger arrangement ved å dobbeltklikke på arrangementet
Respons	Ønsket arrangement blir tilgjengelig for kunden slik at han videre kan velge tilgjengelig dato, antall billetter og dermed setevalg hvis dette er aktuelt for arrangementet.

Use Case	Velge en eller flere billetter
Aktør	Kunde
Forklaring	Kunden kan selv velge antall billetter han ønsker å kjøpe forutsatt at ønsket antall er tilgjengelig for arrangementet.
Stimulus	Kunden har valgt arrangement og lagt inn antall billetter han ønsker å kjøpe.
Respons	Antall billetter valgt blir sjekket opp mot antall billetter tilgjengelig. Hvis antallet er tilgjengelig sendes kunden videre til betaling. Dersom antall billetter valgt er høyere enn antall tilgjengelig blir kunden bedt om å registrere nytt antall.

Use Case	Betale for valgte billetter
Aktør	Kunde
Forklaring	Når kunden har valgt billettene han ønsker, sendes kunden videre til betaling. Her vil kunden få mulighet til å gå gjennom ordren sin, samt velge betalingsløsning og deretter gjennomføre transaksjonen.
Stimulus	Kunden har fylt inn alle nødvendige felt for bestilling av billetter og klikker "Betale/Gå til betaling". Kunden får se detaljer rundt bestillingen og har mulighet til å avbryte/gå tilbake eller gjennomføre kjøpet. Velger gjennomføre kjøpet og registrer nødvendig betalingsinformasjon
Respons	Betalingsinformasjon valideres og transaksjonen gjennomføres hvis valideringen er ok.

Use Case	Søke opp og filtrere arrangementer
Aktør	Kunde
Forklaring	For å sikre best mulig brukeropplevelse må det være mulig å søke opp arrangement basert på arrangement navn, dato, kategori osv. slik at kunden ikke må bla gjennom alle tilgjengelig arrangementer i systemet.
Stimulus	Kunden velger "Søk opp arrangementer" og velger det han vil søke på.
Respons	Arrangementer som svarer til søket vil komme på skjermen. Kunden kan velge aktuelt arrangement for bestilling.

5.3 Brukssituasjoner for kinosjef



Use Case	Registrere spillefilmer
Aktør	Kinosjef og ansatte
Forklaring	For at systemet også skal tilføre verdi for kinosjefen, lar systemet han legge til de spillefilmene han føler er aktuelle. I prinsippet blir filmer lagt til på samme måte som andre arrangementer
Stimulus	Kinosjef registrerer en spillefilm med tilhørende informasjon.(tittel, pris, antall billetter, dato)
Respons	Arrangementet er opprettet og lagt til i liste over forskjellige spillefilmer.

Use Case	Login
Aktør	Kinosjef&ansatte
Forklaring	For at kinosjefen skal få bruke systemet er det påkrevd at man logger inn for å verifisere at han faktisk er kinosjefen. For å logge inn taster han inn e-post adresse og passord
Stimulus	Kinosjef ønsker å logge inn og taster inn innloggingsinformasjon
Respons	Kinosjef er logget inn og kan begynne å bruke systemet.

Use Case	Ta imot betalinger
Aktør	Kinosjef&ansatte
Forklaring	For at kinoen skal kunne motta penger for de solgte billettene sine, skal systemet legge til rette for overføring av penger. Denne funksjonaliteten skal ikke utvikles fra bunnen av, men det skal hentes fra de eksterne leverandørene Stripe og Vipps.
Stimulus	En kunde kjøper en eller flere billetter til en spillefilm.
Respons	Pengene er overført til kinoens konto

Use Case	Overvåke statistikk om filmene
Aktør	Kinosjef&ansatte
Forklaring	Kinosjef/ansatt skal til enhver tid kunne se statistikk over blant annet hvor mange billetter som er solgt.
Stimulus	Kinoen ønsker å se en oversikt over statistikken
Respons	Kinoen blir presentert en oversikt over arrangementets statistikk

Use Case	Presentere kunngjøringer i systemet
Aktør	Kinosjef&ansatte
Forklaring	Å la kinosjef/sjef dele diverse kunngjøringer vil skape større verdi for både kinoen og for kunder. Her vil arrangøren da ha mulighet til å for eksempel øke interessen rundt forskjellige filmer ved å presentere interessante historier som omhandler filmen.
Stimulus	Kinoen skriver en kunngjøring vedrørende arrangementet
Respons	Kunngjøringen blir presentert for kunder.

Use Case	Administrere andre arrangører
Aktør	Kinosjef&ansatte
Forklaring	Noen må inneha rollen som administrator av systemet i forhold til andre arrangører enn kinoene. Det vil si godkjenning av nye arrangører, suspensering av eksisterende arrangører osv.
Stimulus	Arrangør sender forespørsel om å få være arrangør. Kinosjef mottar klage på arrangør el.l.
Respons	Arrangør for tilgang til å opprette brukerkonto. Brukekonto for arrangør blir slettet.

Use Case	Avslutte visningsperiode
Aktør	Kinosjef&ansatte
Forklaring	Etterhvert som en film har vært spilt i en god stund - er det på tide å vise nye filmer på kinoen. Det å la kinoen enkelt avslutte en visningsperiode vil være essensielt viktig.
Stimulus	Kinoen velger å avslutte visningsperioden.
Respons	Filmen er tatt av programmet og det er plass til å registrere en ny film

6. Krav

I denne seksjonen vil du finne krav som er utarbeidet og rettet mot det fullstendig utviklede systemet. Utplukket av de kravene gruppen mener har den viktigste funksjonaliteten prototypen skal inneha - finner du videre i seksjonen som spesifikt omhandler prototypen.

Det er definert to forskjellige typer krav i denne seksjonen: funksjonelle krav og ikke-funksjonelle krav.

Forskjellen på de to kravene er som følger:

- Funksjonelle krav
 - Handler om krav til systemet i form av funksjonalitet som er synlig implementert, det vil si at disse kravene skal vise til hva en bruker av systemet skal kunne gjennomføre.
- Ikke-funksjonelle krav
 - Dette er krav som ikke direkte reflekterer synlig implementert funksjonalitet, men som fortsatt er viktige egenskaper ved systemet. Et eksempel på dette kan være et krav til hvilket programmeringsspråk systemet skal baseres på.

I de tilfeller gruppen ser det nødvendig med en videre forklaring til kravet er det lagt inn et "bullet point" med en mer forklarende tekst til kravet.

6.1 Funksjonelle krav

Kinosjef/administrator av systemet

- 6.1.1 Kinosjef og utvalgte ansatte i administrasjonen til kinoen (eller gruppen med kinoer) skal ha rettighet til å godkjenne nye arrangører.
- 6.1.2 Administrator av systemet skal ha mulighet til å suspendere arrangører som blir tatt i svindel og/eller får gjentatte klager for mangler relatert til tidligere gjennomførte eller ikke gjennomførte arrangementer.

Registrering og innlogging av arrangør

- 6.1.3 Alle arrangører må bli godkjent før de kan begynne å bruke systemet. Ved forespørsel om å opprette arrangør konto må arrangøren oppgi hva slags type arrangementer som i hovedsak vil være aktuelle.
- Det vil alltid være en reell mulighet for at en arrangør setter opp et arrangement som aldri vil finne sted. Det vil si at kunder kan bli utsatt for svindel.
- 6.1.4 Systemet skal la arrangører lage en konto for å logge inn i systemet.
- For at ikke hvem som helst skal kunne opprette arrangementer, må hver arrangør ha en unik bruker i systemet etter godkjenning har funnet sted. Denne arrangør kontoen skal arrangøren logge seg inn med for å legge inn og administrere arrangementer.
- 6.1.5 Systemet gir arrangør muligheten til å registrere bankkonto informasjon til arrangøren, slik at de kan ta imot betalinger.
- For at en arrangør skal kunne motta innbetalinger, må systemet og tilhørende betalingsløsning inkludere informasjon om en typisk «Inn konto» - i likhet med slik arbeidsgiver får fra respektive ansatte.
- 6.1.6. Systemet skal la arrangøren se statistikk på billettstatus på sine arrangement.
- Med billettstatus menes oversikt som viser antall solgte og ledige billetter.

Oppretting av arrangement

- 6.1.7 Systemet skal ha en løsning for å kunne lage et nytt arrangement fra et panel i kontoen til arrangøren.
- Panelet skal lett finnes via en knapp kalt «Opprett arrangement». Brukeren skal så kunne opprette og definere arrangementet etter spesifikke ønsker. Herunder menes det navn, sjanger, aldersgrense, sted, antall billetter o.l.
 - Alle arrangement skal lagres i en database.

Administrasjon av eksisterende arrangement

- 6.1.8 Systemet skal ha en løsning for at arrangøren skal kunne gjøre endringer på et arrangement. Alle attributter til arrangementet skal kunne endres. Dato for arrangement, sted/adresse, antall tilgjengelige billetter osv.
- 6.1.9 Systemet skal ha en løsning for at arrangøren skal ha mulighet til å avlyse arrangementet.
- Ved avlysning skal alle som har kjøpt billett bli varslet på e-mail adressen som bookingbekreftelsen/billetten(e) ble sendt til. Arrangøren skal selv være ansvarlig for at kundene får refundert billett kostnaden.

Registrering og innlogging av kunde

- 6.1.10 Kunder kan velge å opprette kundekonto i systemet.
- Felter som må fylles inn er fornavn, etternavn, e-mail og passord. Input av data i obligatoriske felt skal valideres av systemet før endelig brukerkonto er opprettet. Kundens e-mail adresse skal brukes som brukernavn ved senere innlogging i billettsystemet.
- 6.1.11 Kunder som har opprettet en konto må ha mulighet til å gjøre endringer på kontoinformasjonen.
- 6.1.12 Kunder som har opprettet en konto skal ha mulighet til å slette kontoen sin.
- 6.1.13 Kunder som skal kjøpe billetter kan gjøre dette enten ved å logge inn i systemet, eller ved å fortsette uten å logge inn.

Søke opp og filtrere arrangement

- 6.1.14 Systemet skal la kunden søke etter og filtrere arrangement, slik at de enkelt kan finne fram til riktig arrangement.
- For å la kunden søke etter arrangementer må systemet ha et søkefelt hvor de kan søke etter ønske. I tillegg skal det være mulig å filtrere etter ulike typer arrangement – typisk musikk, kino, idrett etc. Det skal også være mulig å filtrere etter dato og pris. Filtring skal skje via ferdigdefinerte knapper med respektiv informasjon om filtrering på dem.

Bruk av kundens posisjon

- 6.1.15 Systemet skal la kunden tillate at systemet bruker deres posisjon for å fremheve arrangementer i nærheten.
- For at systemet skal kunne bruke kundens posisjon, må kunden godta dette først via en forespørsel fra systemet. Dette er en lettvint måte for kunden å få innblikk i hva som skjer i lokalmiljøet i nærmeste fremtid. Etter et eventuelt kjøp har funnet sted eller kunden forlater systemet, vil ikke systemet fortsette å ha kundens posisjon.

Booking - valg av billett og sete

- 6.1.16 Systemet skal la kunder kjøpe en eller flere billetter til et arrangement.
- 6.1.17 Kunden skal kunne velge hvilket sete han ønsker ut fra hvilke seter som er ledige.

Betaling og verifisering

- 6.1.18 For at kunden skal kunne betale for sine billetter - må følgende være gjort:
- a. En ordreoversikt med et steg for bekreftelse av valgte billetter.

- b. Er kunden en registrert bruker – skal den tilhørende e-postadressen brukes for at kunden mottar billetten. Hvis kunde ikke er en registrert bruker – må kunden manuelt registrere epostadressen for å motta billett.

6.1.19 Systemet skal inneha funksjonalitet som lar kunden betale med betalingskort eller med Vipps.

6.1.20 Kunden skal få bookingbekreftelse når betaling er gjennomført. Booking bekreftelsen sendes sammen med QR-koden(e) til e-postadressen som bookingen ble gjennomført med.

Adgangskontroll

6.1.21 Arrangøren skal ha mulighet til å gjennomføre adgangskontroll.

- Denne kontrollen skal gjøres ved at kunden viser frem QR-koden på billetten, enten på elektronisk enhet eller papir, og at arrangøren scanner denne koden. Scanner enheten må være koblet opp mot systemet og det spesifikke arrangementet.

6.2 Ikke-funksjonelle krav

Kunde

6.2.1 Hvis kunden er inaktiv i mer enn 24 minutter når han/hun skal betale for billetten, så skal sesjonen gå ut og transaksjonen avbrytes.

- Meningen med dette er at billetter ikke skal holdes igjen når en bruker er inaktiv, slik at andre kunder kan kjøpe billetten istedenfor.

6.2.2 Systemet skal hjelpe kunde og arrangør til å fylle inn korrekt info i alle felter.

- Systemet skal da fylle de forskjellige feltene med relevant veiledning for hva kunden/arrangøren skal skrive inn. Et eksempel på dette er i feltet hvor kunden skal skrive inn arrangementsdato. Der vil det stå hvilket format kunden skal skrive inn datoen. Dette vil se slik ut "dd-mm-yy"

6.2.3 Når kunden søker etter et arrangement, så skal søkeresultatene være tilgjengelige for kunden i løpet av 2 sekunder.

- Systemet må være raskt og stabilt, slik at kundene ikke opplever irritasjon ved bruk av systemet.

6.2.4 Hver registrerte brukerkonto må være unik.

- Systemet skal sørge for at det ikke oppstår duplikate e-postadresser. I tillegg skal hver bruker som oppretter en konto bli tildelt et unikt identifikasjonsnummer.

Arrangør

6.2.5 Hver registrerte arrangør må være unik.

- Systemet skal sørge for at det ikke oppstår duplikate arrangører. Det skal kunne kobles flere e-postadresser opp mot en arrangør. I tillegg skal hver arrangør bli tildelt et unikt identifikasjonsnummer.

Arrangement og betaling

6.2.6 Alle arrangement blir tilegnet en unik identifikasjon.

- Systemet skal sørge for at det ikke opprettes duplikater når en arrangør oppretter arrangementer. Det betyr at ingen arrangement skal få lov til å være like i databasen.

6.2.7 Hver billett som genereres, skal få en unik QR-kode.

- Betalingsløsninger skal implementeres ved bruk de eksterne leverandørene Stripe og Vipps.
 - Betalingsløsningene skal gjøre det mulig for kunden å betale med:
 - Kort - Visa og Mastercard
 - Vipps

Databehandling og plattformer

6.2.8 Systemet skal følge gjeldende lover og regler for GDPR

- Personinfo som lagres i systemet knyttet mot brukerkontoer skal lagres og forvaltes i henhold til gjeldende lovgivning med GDPR. Et eksempel er at kunder skal til enhver tid ha tilgang på informasjon om hva systemet bruker deres lagrede informasjon til.

6.2.9 Systemet skal være en Webapplikasjon som skal fungere på alle moderne nettlesere på PC og mobil.

- Dette betyr at systemet skal kunne brukes på nettleserne Chrome, Firefox, Safari og Edge, både på PC og mobil, uten at noe funksjonalitet blir borte.

6.2.10 Systemet skal ha minimum 99% oppetid i løpet av et år.

- For å forhindre tap i form av salg av billetter, så må systemet ha en garantert oppetid på minst 99%, slik at man unngår stor tap i salgsinntektene.

6.2.11 Systemet skal følge alle reglene gitt under "Forskrift om universell utforming av informasjons- og kommunikasjonsteknologiske (IKT)-løsninger".

- Denne forskriften er noe alle IKT-systemer i Norge må følge, så det er derfor kritisk at systemet følger alle punkter i forskriften.

6.3 Estimering av krav til hovedsystemet

Under ligger det vedlagt en liste over kravene til systemet med tilhørende estimat til hvert enkelt krav. Kravene er sortert etter nettoverdien til hvert krav.

6.3.1 - Funksjonelle krav

Krav	Feature	Utviklingstid	Viktighet	Nettoverdi
Funksjonelle				
6.1.5	La arrangør ta imot betaling	Small	X-large	7
6.1.21	Adgangskontroll	Medium	X-large	6
6.1.18	Verifisering av ordre	Medium	X-large	6
6.1.16	La kunde kjøpe billetter	Medium	X-large	6
6.1.19	La kunde betale	Large	X-large	4
6.1.17	Settevalg	Large	X-large	4
6.1.14	Søke opp / filtrere arrangement	Large	X-large	4
6.1.6	La arrangør overvåke statistikk	Small	Medium	3
6.1.20	Sende bookingbekreftelse til registrert kunde epost	Small	Large	3
6.1.2	Suspendere arrangører	Small	Large	3
6.1.12	La kunde slette sin konto	Small	Large	3
6.1.4	Opprette arrangørkonto	Medium	Large	2
6.1.3	Forespørsel om arrangørkonto	Medium	Large	2
6.1.1	Godkjenne nye arrangører	Medium	Large	2
6.1.9	Avlyse arrangement	Small	Medium	1
6.1.8	Endre arrangement	Small	Medium	1
6.1.7	Opprette arrangement	Large	Large	0
6.1.13	Foreta kjøp som registrert bruker eller besøkende	Medium	Medium	0
6.1.11	La kunde gjøre endringer på sin konto	Small	Small	0
6.1.10	Opprette kundekonto	Medium	Small	-1
6.1.15	Bruk av kundens posisjon	Large	Small	-3

6.3.2 - Ikke funksjonelle krav

Ikke-funksjonelle	Feature	Utviklingstid	Viktighet	Nettoverdi
6.2.5	Unik arrangørkonto	Small	X-large	7
6.2.11	Følge universell utforming	Small	X-large	7
6.2.8	Følge lovgivning i henhold til GDPR	Large	X-large	4
6.2.4	Unik brukerkonto	Small	Large	3
6.2.6	Unik arrangement ID	Small	Large	3
6.2.3	Kjapp respons på søk	Medium	Large	2
6.2.7	Generere unik QR kode per billett	Medium	Large	2
6.2.2	Hjelpe med korrekt input	Small	Medium	1
6.2.1	La sesjon utgå etter 24 minutter	Medium	Medium	0
6.2.9	Webapplikasjon - fungere på moderne nettlesere	Large	Large	0
6.2.10	Minimum 99% oppetid/år	Small	Large	0

7. Prototypen

Prototypen av billett applikasjonen inneholder tilstrekkelig funksjonalitet ut ifra det gruppen mener er den viktigste delen av systemet (Minimum Viable Product). Med prototypens viktigste funksjonalitet som grunnlag, er det dokumentert dens begrensninger, implementerte krav (med tilhørende tester og estimering av utviklings omfang i henhold til forretningsnytte) og sekvens- og aktivitetsdiagrammer.

Under seksjonen for kravene til prototypen vil du finne selve kravet, med tilhørende tester for å utelukke feil i koden. I de tilfeller gruppen føler det er nyttig med en ytterligere forklarende tekst til kravet, vil dette bli presentert i et "bullet point" rett under kravet.

7.1 Krav for å kjøre prototypen

For å unngå potensielle problemer ved bruk av prototypen, er det anbefalt at man har følgende verktøy og versjoner installert:

- Java versjon 11
- Junit versjon 5
- Maven versjon 3.3.9

Med disse verktøyene og versjonene installert skal det ikke være noe problem å kjøre prototypen.

7.2 Begrensninger og antagelser for prototypen

- GDPR
 - Implementasjon og hensyn til GDPR lovgivning vil i prosjektperioden - og utviklingen av prototypen bli sett bort ifra. Men det skal poengteres at det endelige produktet skal utvikles i henhold til gjeldende GDPR's lovgivning og standarder, slik at kunders personvern blir vernet. Et eksempel hvor GDPR spesielt skal bli tatt høyde for er i krav 1.1.1 - som dreier seg om registrering av arrangører med respektiv personinformasjon/bedriftsinformasjon.
- Brukergrensesnitt og universell utforming
 - Brukergrensesnitt og universell utforming er også aspekter ved prototypen som ikke vil bli lagt vekt på i prosjektperioden. Gruppen har som felles interesse å heller legge vekt på kjernefunksjonaliteten til systemet, i stedet for å rette fokus på utseende og brukervennligheten til prototypen av systemet.

- Betalingsløsning
 - Gruppen skal i prototypen ikke implementere en fungerende betalingsløsning, men for å vise fram hvordan betaling kan fungere, så skal det lages en falsk betalingsløsning. Denne vil være veldig enkel, og skal kun være et eksempel på hvordan betalingen skal kunne gjennomføres og hvordan feilmeldinger kan vises. Dette vil typisk være at et kortnummer som starter på tallet "4" ikke har nok midler til å betale og kort som starter på tallet "5" er kort som ikke er godkjent.
- Lagring av data
 - Lagring av data vil i prosjektperioden skje i minne på maskinen. Dette er pga det ikke vil være nødvendig å lagre masse data når man skal vise hvordan en prototype av et system vil fungere. Den endelige utgaven av systemet vil lagre informasjon i respektive databaser.
- Betaling for kunder
 - I hovedkravene har vi nevnt at kunder skal kunne betale uten å måtte lage en brukerkonto. I prototypen har vi derimot valgt å gjøre det slik at kunden må lage en brukerkonto for å kjøpe billetter. Grunnen til dette er at det skal være et tydelig skille på funksjonalitet til kunder, og funksjonaliteten til arrangører. Siden denne prototypen ikke har et ordentlig brukergrensesnitt kunne det derfor bli vanskelig å skille mellom de forskjellige funksjonalitetene.
- Implementasjon av krav
 - Kravene som er implementert i prototypen vil i noen tilfeller være noe modifisert. Grunnlaget for dette er gruppens kompetanse innenfor fagfeltet som naturlig setter noen begrensninger i evnen til å implementere alt som er ønsket og tiltenkt funksjonalitet i hovedsystemet.
 - Andre faktorer som spiller inn er hva gruppen mener er nødvendig funksjonalitet for en prototype med tanke på at det skal leveres et "Minimum Viable Product".
 - Et eksempel på lett modifisering er i kravene til hovedsystemet, skal en kunde kunne gjennomføre kjøp av uten å være logget inn i systemet som bruker. I prototypen er det implementert funksjonalitet som gjøre at en kunde må være innlogget.

7.3 Oversiktstabell over krav og viktighet for prototypen

Under er det vedlagt en oversiktstabell over de kravene/feature som gruppen har betraktet som de viktigste i henhold til leveranse av en fungerende prototype. Det vil forekomme avvik fra estimater som er å finne i tabellen over "krav til hovedfunksjonalitet". Grunnen til dette er at krav som er essensielle for at et komplett system skal fungere som det er forventet å gjøre - ikke vil være like viktig for å demonstrere flyten i en prototype.

Feature	Utviklingstid	Viktighet	Nettoverdi
Søke opp / filtrere arrangement	Small	X-large	7
Betaling for billetter	Medium	X-large	6
Opprette kundekonto	Medium	X-large	6
Logg inn for kunde	Medium	X-large	6
Booke arrangement	Large	X-large	4
Opprette arrangørkonto	Medium	Medium	0
Logg inn for arrangør	Medium	Medium	0
Adgangskontroll	X-large	X-large	0
Opprette arrangement	Large	Medium	-2

7.4 Forklaring til krav og deres viktighet

Følgende punkter er de vi i gruppen mener er det viktigste:

- Søke opp/filtrere arrangement
 - Dette kravet er ansett som et av de viktigste med tanke på brukeropplevelsen til kunden. For å ikke miste kundens tålmodighet ved at han eller hun må bla gjennom hver eneste arrangement, så er det nødvendig med en god søkemotor som gjør at kunden finner det han/hun ønsker. For at man skal kunne selge billetter til arrangementer, så er søk av arrangement svært viktig.
- Betaling for billetter
 - Dette er essensiell funksjonalitet for at systemet skal tilføre verdi både for kunde og for arrangør - ettersom systemet skal legge til rette for kjøp og salg av billetter til arrangementer.
- Opprette kundekonto
 - I vårt tilfelle med slik prototypen er utformet, vil det ikke være mulig å demonstrere hendelsesforløpet i et kjøp uten at den som skal kjøpe billetten på forhånd har registrert en brukerkonto og er innlogget i systemet.
- Innlogging for kunde
 - Se kommentar for "opprette kundekonto".

- Booke arrangement
 - Dette er funksjonalitet som både tilfører verdi for kunde og for arrangør. Sett i sammenheng med den tiltenkte hovedfunksjonaliteten til systemet, er dette funksjonalitet som er meget essensielt for et billettsystem.

Følgende punkter er ansett som mindre viktige deler av systemet:

- Opprette arrangør-konto og innlogging for arrangør.
 - Etter gruppens mening om hva som er den viktigste funksjonaliteten til prototypen og for å være et "Minimum Viable Product", er ikke dette funksjonalitet som klassifiseres som den viktigste. Begrunnelsen er dette kravet bli gjort manuelt i startfasen når man tester hvordan systemet fungerer i praksis. I prototypen har vi valgt å ha med en enkel implementasjon av dette som er nesten helt lik oppretting av kundekonto.
- Opprette arrangement
 - Som nevnt i forrige punkt, er ikke dette funksjonalitet som klassifiseres som den viktigste. Begrunnelsen er dette arrangement kan opprettes manuelt i startfasen når man tester hvordan systemet fungerer i praksis.

7.5 Funksjonelle krav implementert i prototypen

Under er det dokumentert hvilke krav som er implementert i prototypen av systemet utfra det prosjektgruppen mener er den viktigste funksjonaliteten. I de tilfeller gruppen ser det nødvendig med en ytterligere forklaring til kravet, er dette lagt inn i et "bullett point".

I tillegg til hvert enkelt krav, er det også dokumentert tester tilhørende de respektive kravene. Testene er dokumentert med en test ID, test navn og i hvilken fil testen ligger. For å ikke gjøre det for uoversiktlig, har vi valgt å ikke vise til alle testene her, bare de testene som viser at funksjonaliteten er der. For en full oversikt over alle tester, besøk Readme-filen til prosjektet.

7.5.1 Registrering og innlogging av arrangør.

Krav 6.1.4: *Systemet skal la arrangører lage en konto for å logge inn i systemet.*

For at arrangøren skal registrere seg, må systemet ha fornavn, etternavn, firmanavn, epost og passord for å kunne opprette kontoen. Hvis e-postadressen allerede er registrert vil det komme opp en melding om dette og arrangøren vil få mulighet til å registrere seg på nytt.

- Test ID og navn
 - 5H: checkIfOrganizerIsAddedToList()
 - 5J: checkIfOrganizerExistsTestReturnsTrue()
- Fil: NewUserAccountTest

Etter at arrangøren har registrert kontoen sin, har han/hun mulighet til å logge inn ved å skrive inn e-postadressen og passord angitt ved registrering av konto.

- Test ID og navn
 - 4A: `checkIfOrganizerLoginReturnsObjectIfFound()`
- Fil: `LoginOrganizerTest`

Krav 6.1.6: *Systemet skal la arrangøren se statistikk på billettstatus på sine arrangement.*

Med billettstatus menes oversikt som viser antall solgte og ledige billetter.

- Test ID og navn:
 - 2G: `checkIfEventStatisticsWorks()`
- Fil: `EventManagerTest`

7.5.2 Oppretting av arrangement

Krav 6.1.7: *Systemet skal ha en løsning for å kunne lage et nytt arrangement fra et panel i kontoen til arrangøren.*

- Test ID og navn
 - 2B: `checkIfEventsAddedToListWhenCreateEventMethodIsCalled()`
- Fil: `EventManagerTest`

7.5.3 Registrering og innlogging av kunde

Krav 6.1.10: *Kunder kan velge å opprette kundekonto i systemet.*

Felter som må fylles inn er fornavn, etternavn, e-mail og passord.

Hvis e-mail adressen allerede er registrert vil det komme opp en melding om dette og brukeren vil få mulighet til å registrere seg på nytt.

- Test ID og navn:
 - 5G: `checkIfUserIsAddedToList()`
 - 5I: `checkIfCustomerExistsTestReturnsTrue()`
- Fil: `NewUserAccountTest`

Etter at kunden har registrert kontoen sin, har han/hun mulighet til å logge inn ved å skrive inn e-postadressen og passord angitt ved registrering av konto.

- Test ID og navn
 - 5A: `checkIfCustomerLoginReturnsObjectIfFound()`
- Fil: `LoginCustomerTest`

7.5.4 Søke opp og filtrere arrangement

Krav 6.1.14: *Systemet skal la kunden søke etter og filtrere arrangement, slik at de enkelt kan finne fram til riktig arrangement.*

Kunden kan søke etter arrangement ved å velge søk etter arrangement. I prototypen er det lagt opp til at kunden kan søke på enten arrangement eller kategori.

- Test ID og navn
 - 7A: `checkIfSearchEventByEventNameReturnsArrayOfEventObjects()`
 - 7C: `checkIfSearchEventByCategoryNameReturnsArrayOfEventObject()`
- Fil: `SearchEventTest`

7.5.5 Booking - valg av billett og sete

Krav 6.1.16: *Systemet skal la kunder kjøpe en eller flere billetter til et arrangement.*

- Test ID og navn:
 - 1B: `checkIfTicketAmountDecreasesProperly()`
 - 1D: `checkIfTicketIsAddedToSoldTicketsListWhenPaymentIsAccepted()`
- Fil: `BookingTest`

7.5.6 Betaling og verifisering

Krav 6.1.19: *Systemet skal inneha funksjonalitet som lar kunden betale med betalingskort eller med Vipps.*

Kunden skal registrere kortnummer, utløpsdato og CVC kode. Betalingsinfo blir delvis kontrollert, men er ikke koblet opp til Stripe. I prototypen vil du kun ha mulighet til å betale med kort.

- Test ID og navn:
 - 6A: `checkIfCardNumberValidationWorks()`
 - 6D: `checkIfCVCMMethodWorks()`
 - 6E: `checkIfDateCheckMethodWorks()`
- Filnavn: `PaymentTest`

7.5.7 Adgangskontroll

Krav 6.1.21: *Arrangøren skal ha mulighet til å gjennomføre adgangskontroll.*

Arrangøren skal ha mulighet til å sjekke om en billett er brukt opp, og han/hun skal ha mulighet til å markere en billett som brukt.

- Test ID og navn:
 - 2A: `checkIfMarkTicketMethodWorks()`
 - 2B: `checkIfTicketControlWorks()`
- Fil: `EventManagerTest`

7.6 Ikke funksjonelle krav implementert i prototypen

7.6.1 Arrangement og betaling

Krav 6.2.6: *Alle arrangement blir tilegnet en unik identifikasjon.*

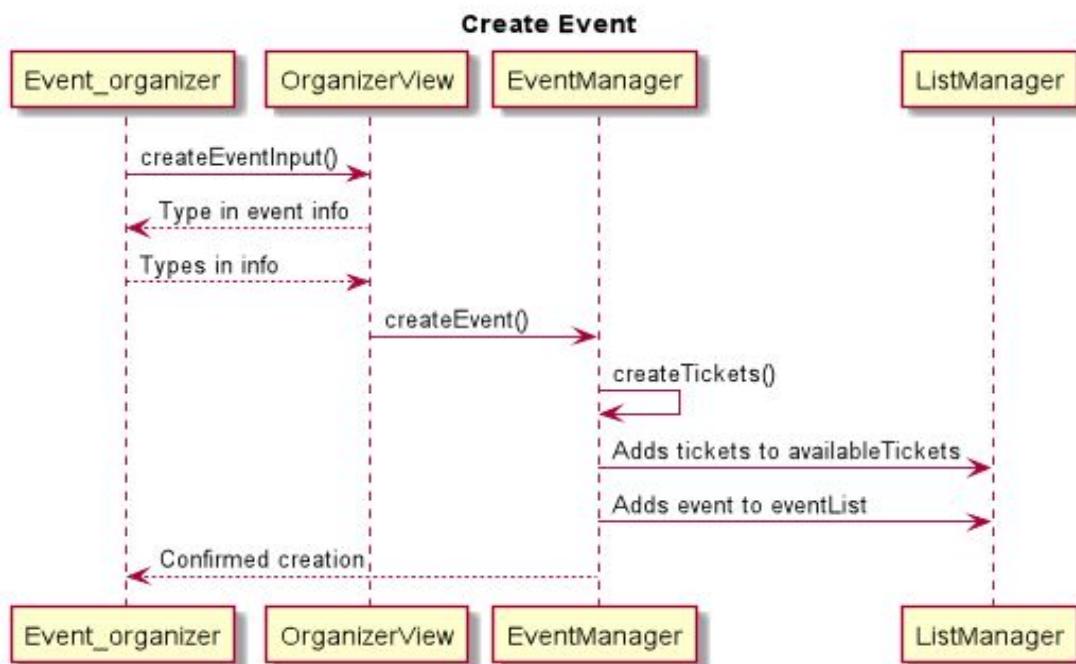
Systemet skal sørge for at det ikke opprettes duplikater når en arrangør oppretter arrangementer.

- Test ID og navn:
 - 2C: `checkIfCreateIndexWorks()`
- Filnavn: `EventManagerTest`

8. Sekvensdiagram

Under er det vedlagt forskjellige sekvensdiagrammer som omhandler forskjellig funksjonalitet i prototypen. Hvert sekvensdiagram illustrerer hvordan systemet oppfører seg i ulike tidssekvenser og hvordan de forskjellige objektene og klassene som er involvert i samme scenario samarbeider for å løse en gitt oppgave.

8.1 Oppretting av arrangement

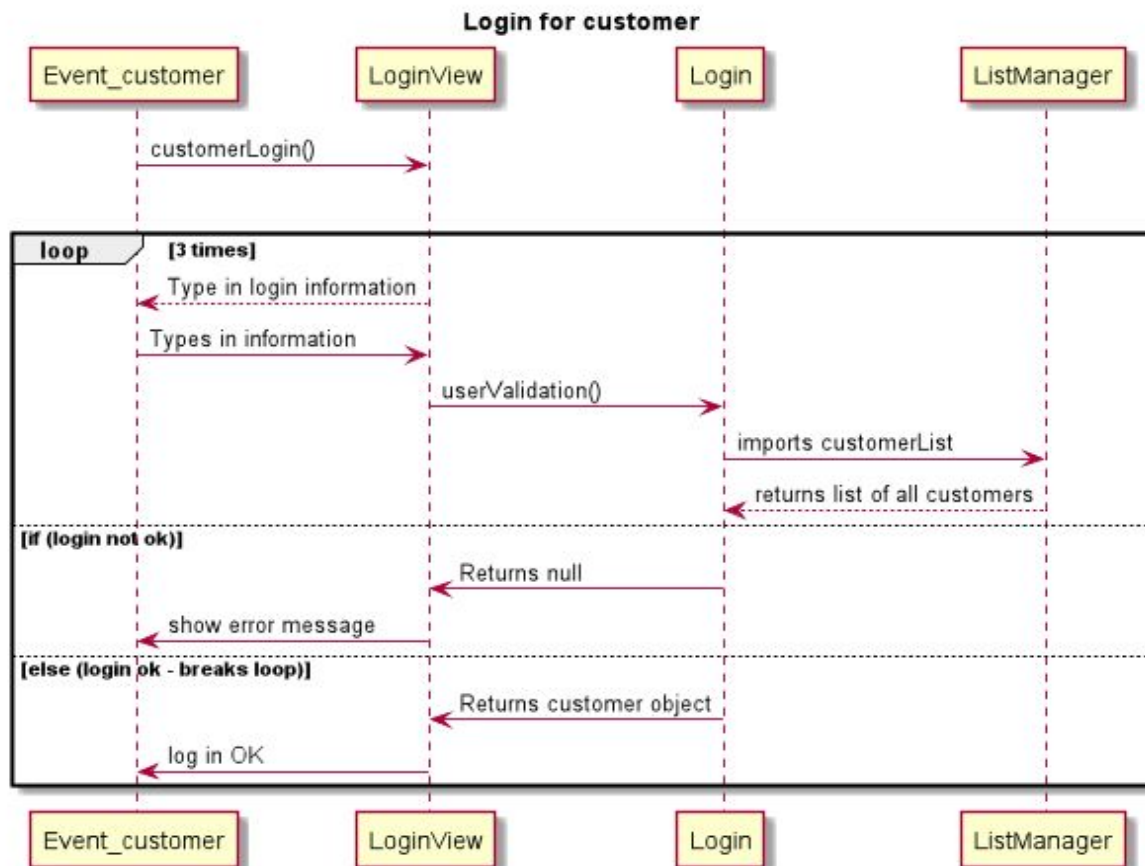


Modellen over (8.1) beskriver hvordan systemet oppretter et nytt arrangement.

Event_organizer refererer til arrangøren i dette tilfellet. Det første arrangøren møter når han/hun skal opprette en nytt arrangement er OrganizerView. Systemet vil komme med flere valg for arrangøren om hva hun/han vil gjøre, men i dette tilfellet er det å lage et nytt arrangement.

OrganizerView-klassen vil spørre om den nødvendige informasjonen for å lage arrangement, for deretter å sende denne informasjonen til validering. Hvis valideringen ikke blir godkjent, vil brukeren bli spurt om å fylle inn informasjonen på nytt. Når all validering er godkjent vil klassen kalle på `createEvent()` funksjonen som ligger i EventManager-klassen, den funksjonen igjen kaller på en annen (`createTickets()`) for å lage billetter for arrangementet. Når alle objektene har blitt laget vil de bli lagt til i sine riktige lister i ListManager-klassen.

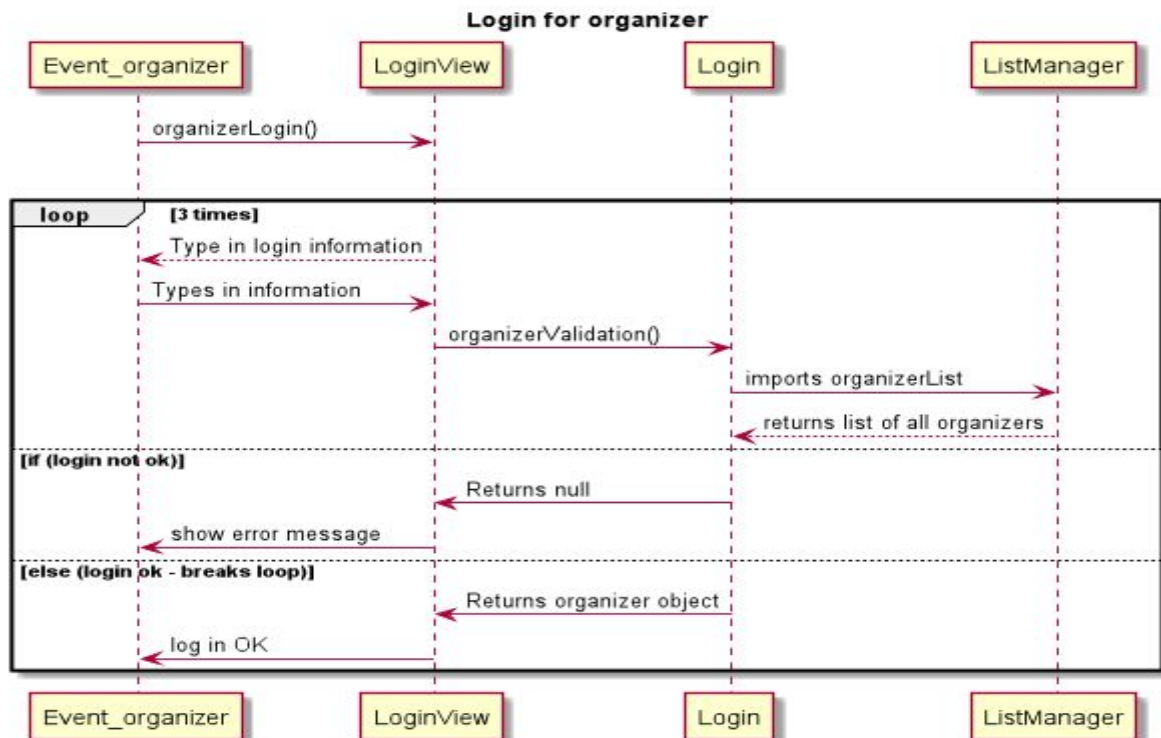
8.2 Innlogging for kunder



Modellen ovenfor (8.2) beskriver hvordan innloggingen for kunder i prototypen fungerer. Event_customer referer i dette tilfellet til kunden. Det første kunden møter når han/hun skal logge inn er klassen LoginView, som er brukergrensesnittet. Denne spør om innloggingsinformasjon.

Informasjonen som blir skrevet inn av brukeren blir deretter sendt videre til Login-klassen som henter listen med alle kunder. Login-klassen sjekker deretter om den spesifikke epostadressen og det spesifikke passordet eksisterer i listen. Hvis den finner en match, så blir kunden innlogget. Hvis det ikke matcher, så får brukeren opp en feilmelding. Det er maks tre forsøk, så hvis man feiler den tredje gangen blir man sendt tilbake til forsiden av prototypen.

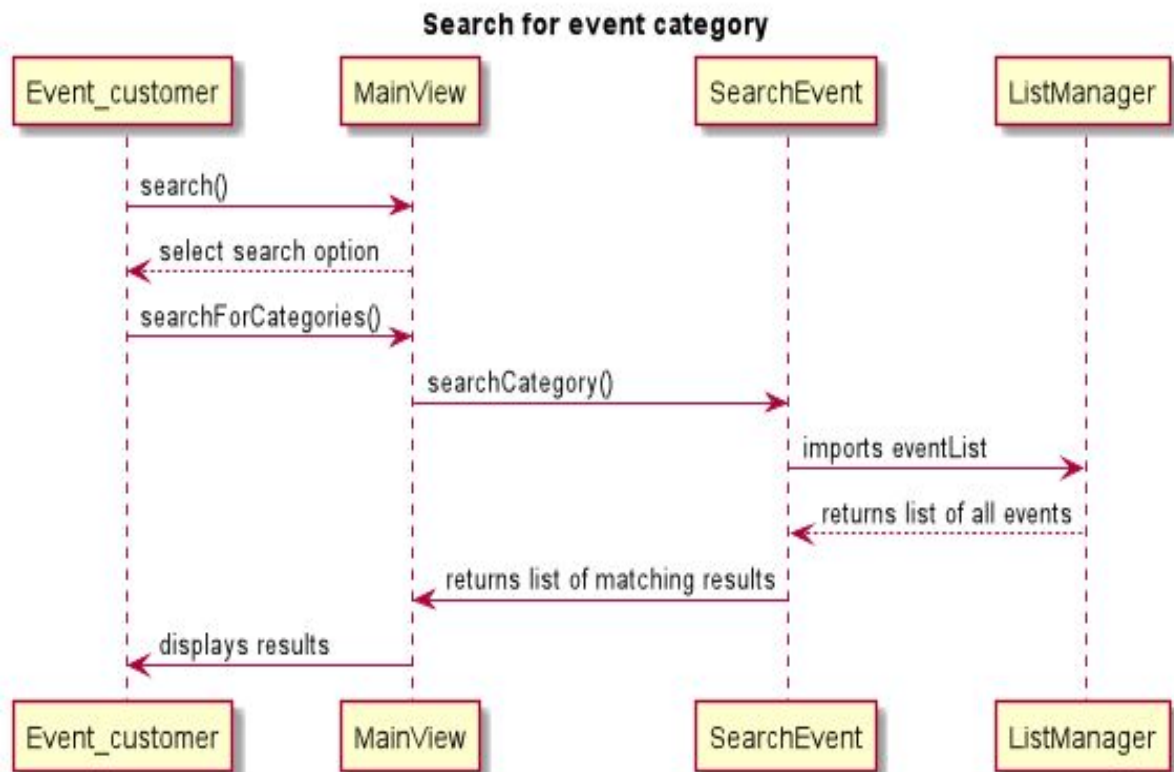
8.3 Innlogging for arrangører



Modellen ovenfor (8.3) beskriver hvordan innloggingen for arrangører i prototypen fungerer. Event_organizer referer i dette tilfellet til arrangøren. Det første arrangøren møter når han/hun skal logge inn er klassen LoginView, som er brukergrensesnittet. Denne spør om innloggingsinformasjon.

Informasjonen som blir skrevet inn av brukeren blir deretter sendt videre til Login-klassen som henter listen med alle arrangører. Login-klassen sjekker deretter om den spesifikke epostadressen og det spesifikke passordet eksisterer i listen. Hvis den finner en match, så blir arrangøren innlogget. Hvis det ikke matcher, så får brukeren opp en feilmelding. Det er maks tre forsøk, så hvis man feiler den tredje gangen blir man sendt tilbake til forsiden av prototypen.

8.4 Søk etter kategori

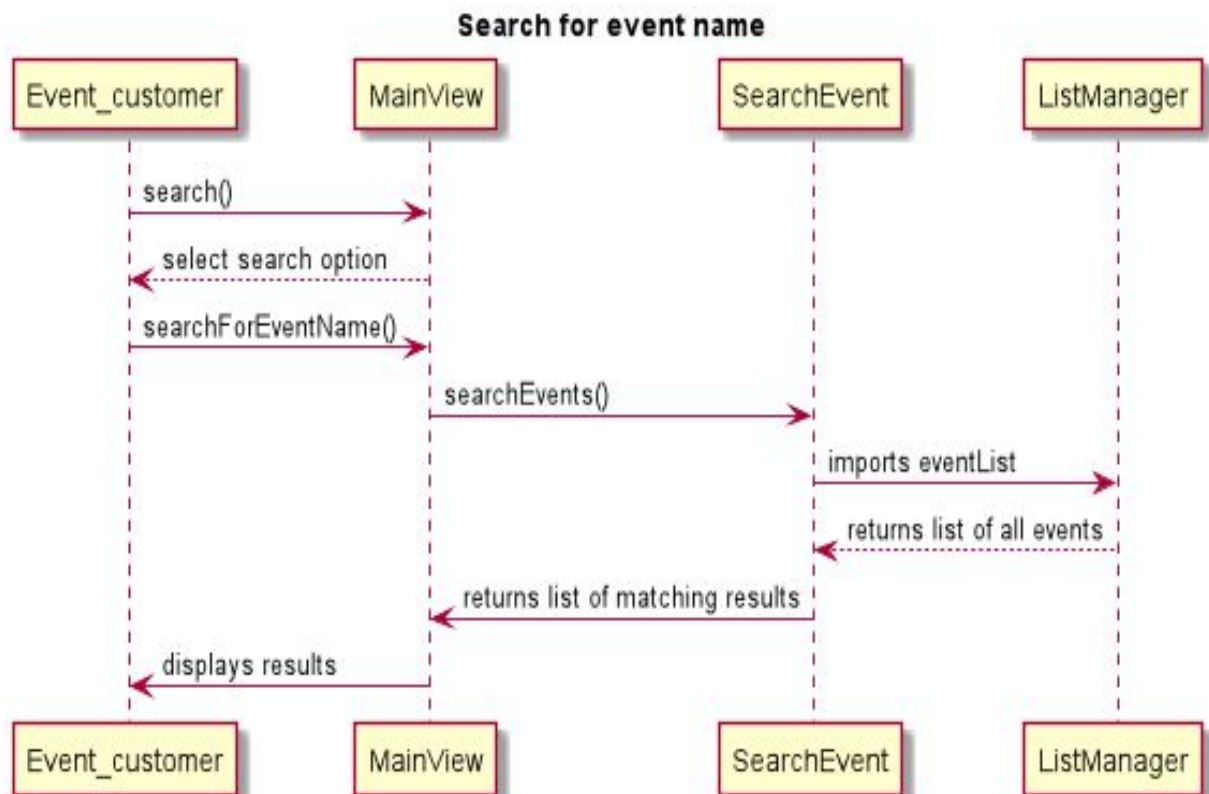


Modellen ovenfor (8.4) beskriver hvordan søk av kategori fungerer i prototypen.

Event_customer referer i dette tilfellet til kunden. Det første kunden møter er MainView, som er brukergrensesnittet. Her får kunden mulighet til å velge to mellom alternativer for søk, som er "Søk etter kategori" eller "Søk etter arrangement".

Hvis bruker velger "Søk etter kategori" blir søkeordet brukeren skriver inn sendt til SearchEvent-klassen. Denne går gjennom listen med alle arrangement, og ser om det finnes noen kategorier som ligner på det brukeren søkte etter. Tilslutt blir en liste med alle matchende arrangement som har den kategorien sendt tilbake til brukeren og vist fram.

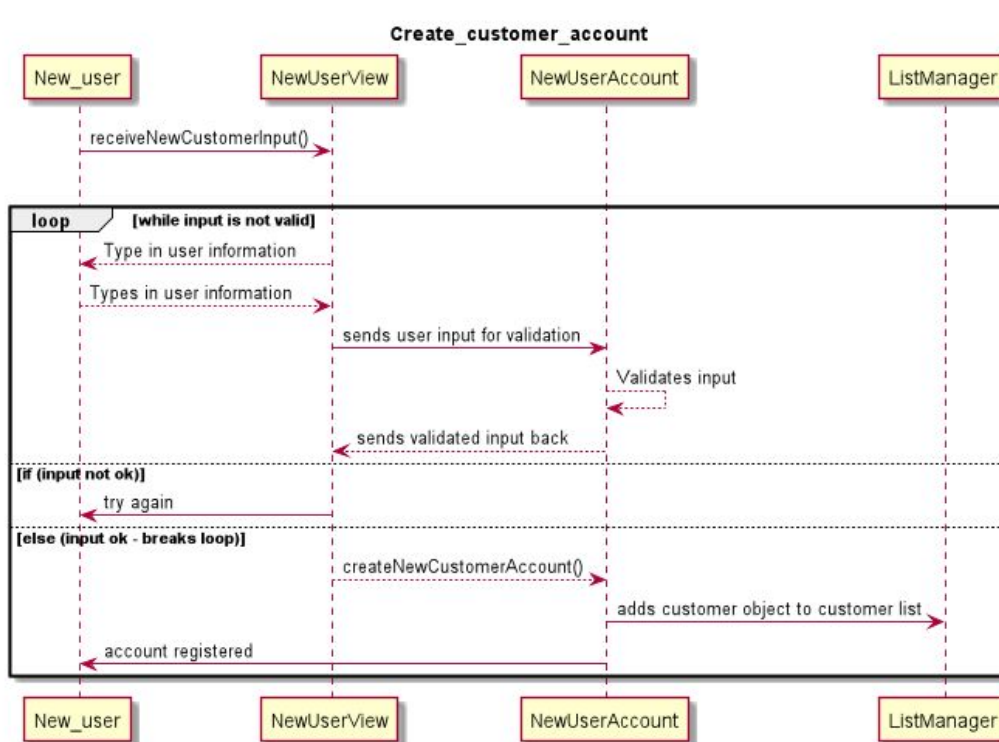
8.5 Søk etter arrangement



Modellen ovenfor (8.5) beskriver hvordan søk av arrangement fungerer i prototypen. Event_customer referer i dette tilfellet til kunden. Det første kunden møter er MainView, som er brukergrensesnittet. Her får kunden mulighet til å velge to mellom alternativer for søk, som er "Søk etter kategori" eller "Søk etter arrangement".

Hvis bruker velger "Søk etter arrangement", blir søkeordet brukeren skriver inn sendt til SearchEvent-klassen. Denne går gjennom listen med alle arrangement, og ser om det finnes noen arrangement som ligner på det brukeren søkte etter. Tilslutt blir en liste med alle matchende arrangement sendt tilbake til brukeren og vist fram.

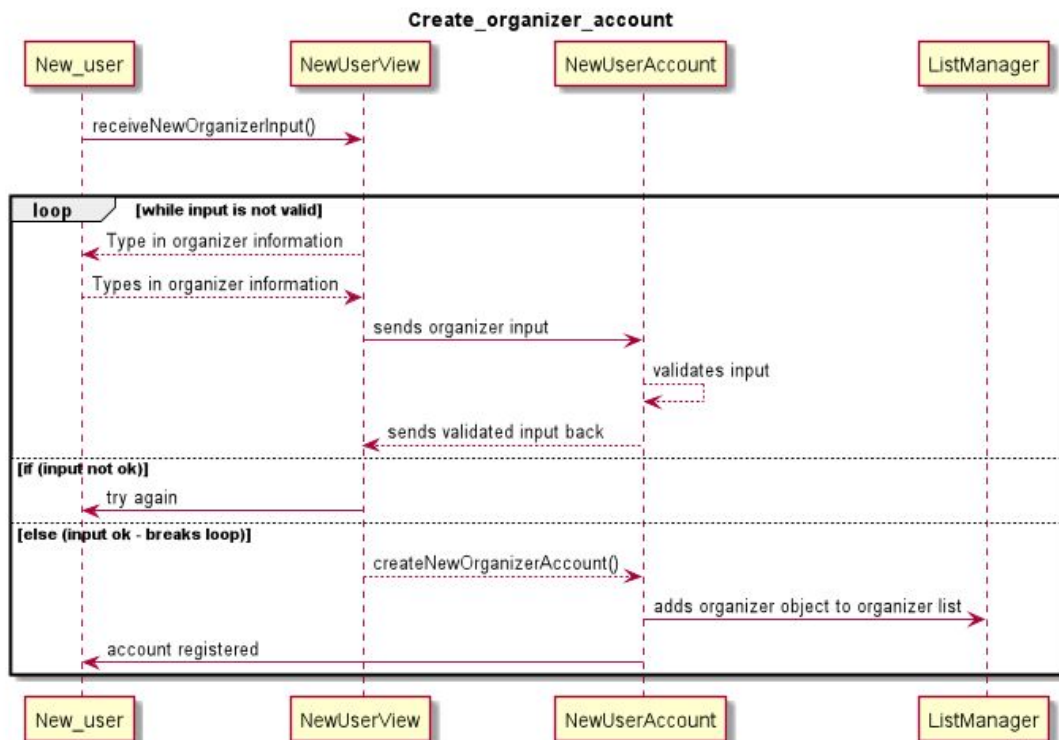
8.6 Opprette kundekonto



Diagrammet over (8.6) beskriver prosessen som skjer når en ny kunde-bruker opprettes i systemet. Kunden skriver inn informasjonen sin: fornavn, etternavn, e-post og ønsket passord. For hver ting brukeren skriver inn blir inputen sendt til metoder i NewUserAccount som validerer dem, hvis valideringen ikke er ok må brukeren skrive inn ny input. Denne prosessen skjer på nytt helt til inputen er godkjent.

Den ferdig validerte informasjonen sendes tilbake til NewUIView som oppretter et nytt kundeobjekt og sender det til metoden createNewOrganizerAccount() i NewUserAccount. Denne metoden sjekker om kunden allerede eksisterer, hvis den ikke gjør det blir det nye kundeobjektet lagt til i customerList i klassen ListManager. Når kunde-brukeren er opprettet får brukeren beskjed om dette.

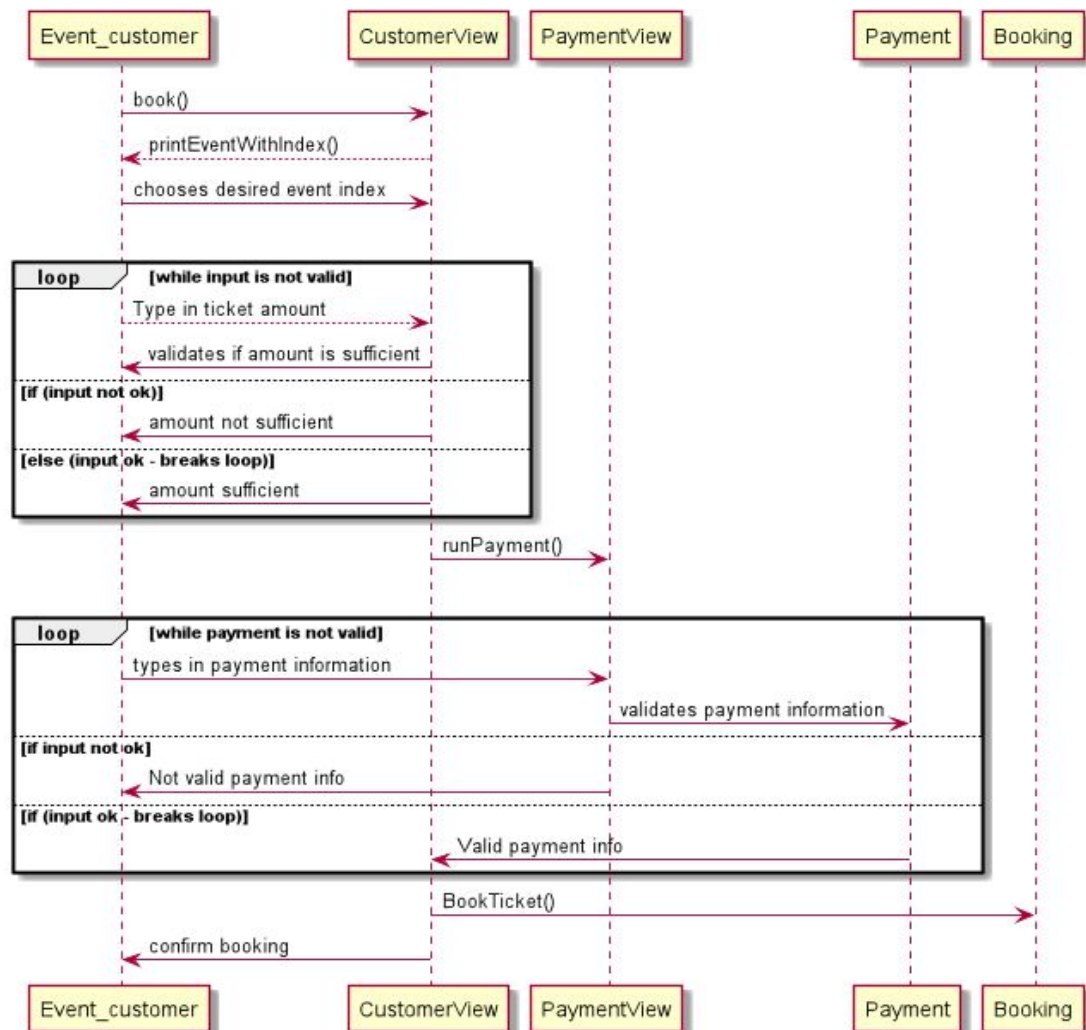
8.7 Opprette arrangør konto



Diagrammet over (8.7) beskriver prosessen som skjer når en ny arrangør-bruker opprettes i systemet. Arrangøren skriver inn informasjonen sin: firmanavn, fornavn, etternavn, e-post og ønsket passord. For hver ting brukeren skriver inn blir inputen sendt til metoder i `NewUserAccount` som validerer dem, hvis valideringen ikke er ok må brukeren skrive inn ny input. Denne prosessen skjer på nytt helt til inputen er godkjent.

Den ferdig validerte informasjonen sendes tilbake til `NewUIView` som oppretter et nytt arrangør-objekt og sender det til metoden `createNewOrganizerAccount()` i `NewUserAccount`. Denne metoden sjekker om arrangøren allerede eksisterer, hvis den ikke gjør det blir det nye arrangør-objektet lagt til i `organizerList` i klassen `ListManager`. Når kontoen er opprettet får brukeren beskjed om dette.

8.8 Booke billetter og betaling



Diagrammet over (8.8) beskriver hvordan booking- og betaling av billetter foregår i prototypen. I dette tilfellet er Event_customer en kunde som skal booke og betale for billetter.

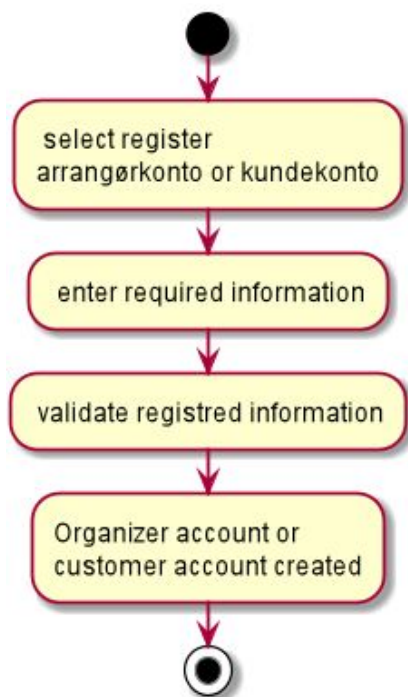
Brukeren blir møtt av CustomerView etter at hun/han har logget inn med sin brukerkonto. Her er det en rekke forskjellige ting brukeren kan velge å gjøre, men i dette tilfellet så tar vi for oss booking. Når brukeren har valgt å booke vil systemet kalle på funksjonen book() som spør brukeren om all nødvendig informasjon om antall billetter han/hun ønsker å kjøpe. Om brukeren velger et ugyldig antall billetter vil systemet si ifra og spørre om ny input. Etter dette vil CustomerView kalle på runPayment i PaymentView, denne funksjonen vil spørre brukeren om en 16 sifret kortnummer, utløpsdato og en 3 sifret CVC, om informasjonen ikke blir validert riktig vil det be om ny input og kjøre validering på nytt. Når betalingsinformasjonen er skrevet inn korrekt vil CustomerView kalle på BookTicket() funksjonen i Booking-klassen som vil sette kundens email på billettene, fjerne de fra availableTickets listen og legge dem

til i soldTickets listen. Deretter vil CustomerViewet sende en melding til brukeren om at bookingen har blitt gjennomført.

9. Aktivitetsdiagram

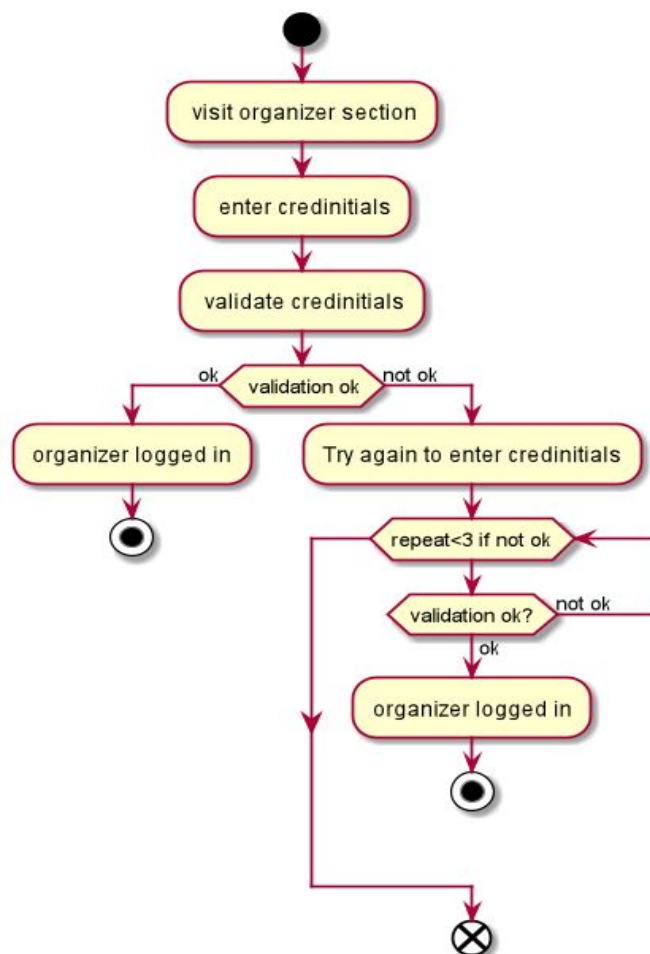
Under er det vedlagt aktivitetsdiagram tilegnet ulik funksjonalitet i prototypen. Disse diagrammene beskriver hvordan en funksjon eller oppgave løses i "virkeligheten", og viser hva som skjer og hvilke avgjørelser som tas.

9.1 Registrere kunde- og arrangør konto



Til venstre viser modellen til hva som skjer i systemet når en kunde eller en arrangør registrerer i systemet. På forsiden til applikasjonen velger man å trykke på "registrer". Videre krever systemet at kunden/arrangøren skriver inn nødvendig informasjon som fullt navn, e-post, passord og firmanavn hvis det er snakk om en arrangør. Etter kunden /arrangøren foretar systemet en validering av det som er skrevet inn. Hvis kravene til informasjon er godkjent av systemet, blir kontoen registrert. Hvis informasjon ikke er godkjent, blir vedkommende bedt om å skrive inn på nytt, helt til informasjonen godkjennes.

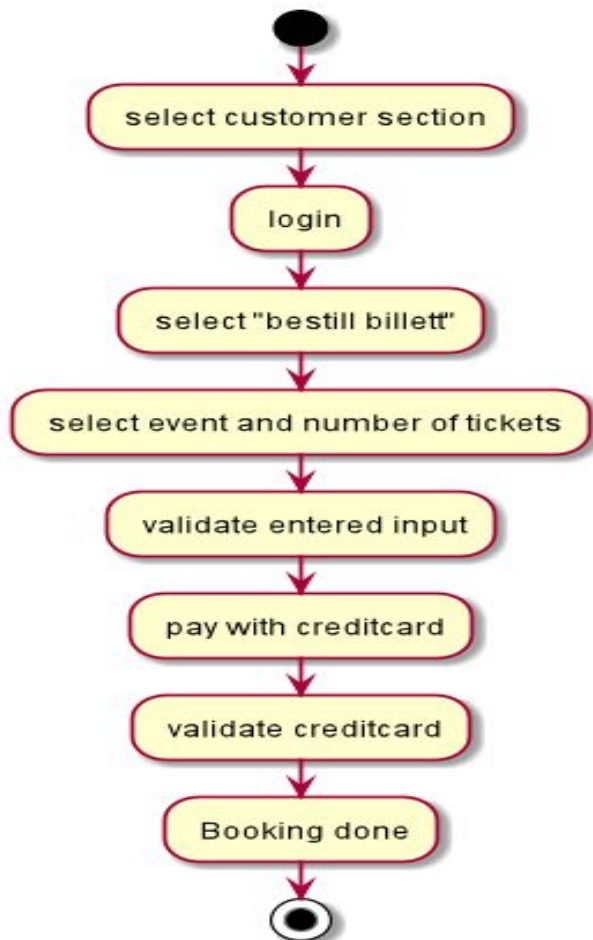
9.2 Innlogging for arrangører og kunder



På forsiden av prototypen så vil brukeren få muligheten til å velge mellom kunde- og arrangør seksjon. For å komme inn på disse seksjonene, må man logge inn med en brukerkonto. Modellen til venstre viser stegene som må til for at brukeren skal bli logget inn. Når brukeren har valgt hvilken seksjon han/hun vil inn på, må brukeren skrive inn e-postadresse og passord. Videre sjekker systemet om brukeren med den e-posten og det passordet eksisterer, og hvis den finner en bruker, så blir brukeren logget inn.

Hvis systemet ikke finner en bruker, vil brukeren få en feilmelding med mulighet til å prøve på nytt. Hvis bruker prøver tre ganger og alle gangene feiler, blir kunden sendt tilbake til forsiden.

9.3 Bestilling og betaling av billetter

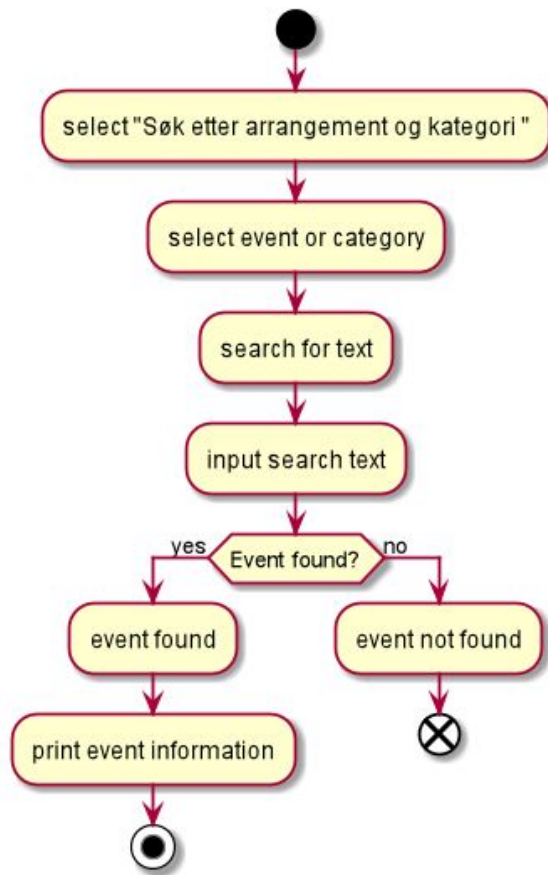


For at kunden skal kunne bestille og betale for billetter, så må kunden være logget inn (i prototypen). På forsiden vil kunden få muligheten til å logge inn ved å velge "Kunde seksjon" alternativet i menyen.

I kunde seksjonen vil brukeren se et alternativ i menyen som heter "Bestill billett". Derfra har kunden mulighet til å velge ønsket arrangement og antall billetter.

Når antall billetter og arrangement har blitt validert, vil kunden få opp en seksjon som tar imot betalingsinformasjon. Her vil kunden bli bedt om kortnummer, CVC og utløpsdato. Når alle disse har blitt validert, vil bestillingen av billetter være gjennomført.

9.4 Søk av arrangement og kategori



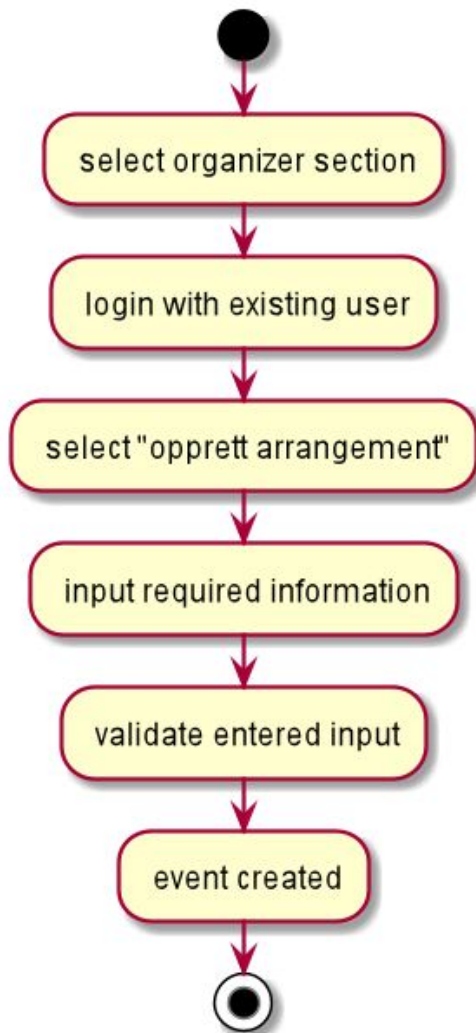
På forsiden av prototypen ligger det en seksjon i menyen som heter "Søk etter arrangement og kategori". Når brukeren går inn på denne, kommer det opp to alternativer "Søk etter arrangement" og "Søk etter kategori". Når valget er tatt kan brukeren begynne å skrive inn søkeord.

Hvis brukeren valgte søk av arrangement vil systemet bruke søkeordet til å se om det finnes noen arrangement som ligner på det brukeren søkte etter. Alle arrangement som matcher vil bli vist.

Hvis brukeren valgte søk av kategori, vil systemet bruke søkeordet til å se etter lignende kategorier. Alle arrangement med den kategorien vil da bli vist.

Tilslutt vil resultatene bli vist til brukeren. Hvis systemet ikke finner noen arrangement, vil brukeren få opp en passende feilmelding.

9.5 Opprette arrangement



På forsiden i prototypen må arrangøren velge "arrangement seksjon". Der vil han videre måtte oppgi login info for å få lov å opprette arrangement.

Når logget inn må han velge "opprett arrangement" fra menyen.

Arrangøren vil da få opp ett og ett spørsmål om arrangementet som må besvares.

All input fra arrangøren vil bli validert og han kommer ikke videre før korrekt input er gitt i henhold til gjeldende validerings kriterier i prototypen.

Når alle felt er fylt inn vil arrangementet bli opprettet i systemet.

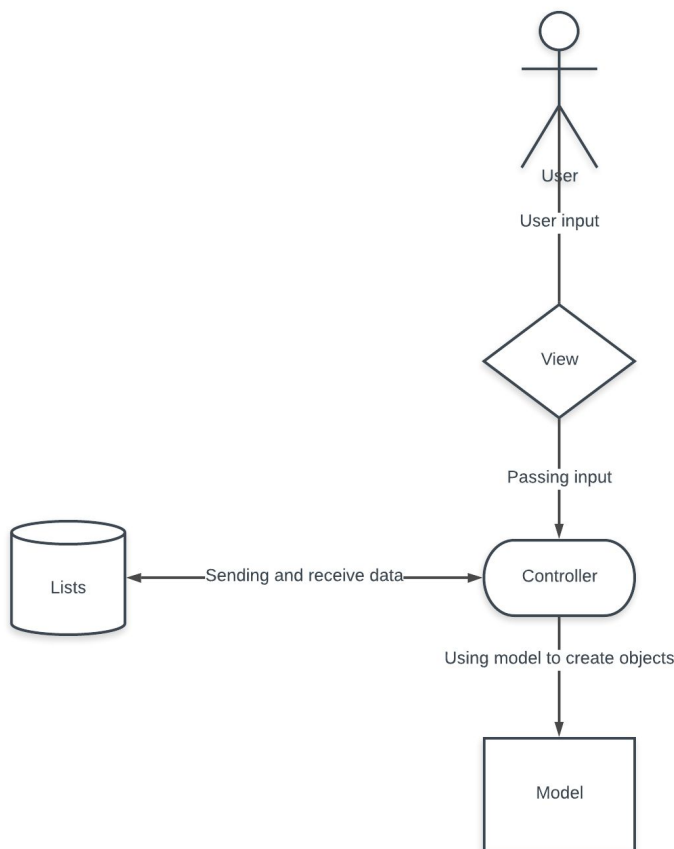
10. Beskrivelse av prototypen

Prosjektet er basert på MVC modellen (Model, View, Controller). Dette har en stor betydning på hvordan de forskjellige delene av systemet kommuniserer med hverandre. Ettersom at vi har valgt å ikke bruke en database for lagring av data i dette prosjektet så er det også en ting å trekke i fra som simplifiserer kommunikasjonen.

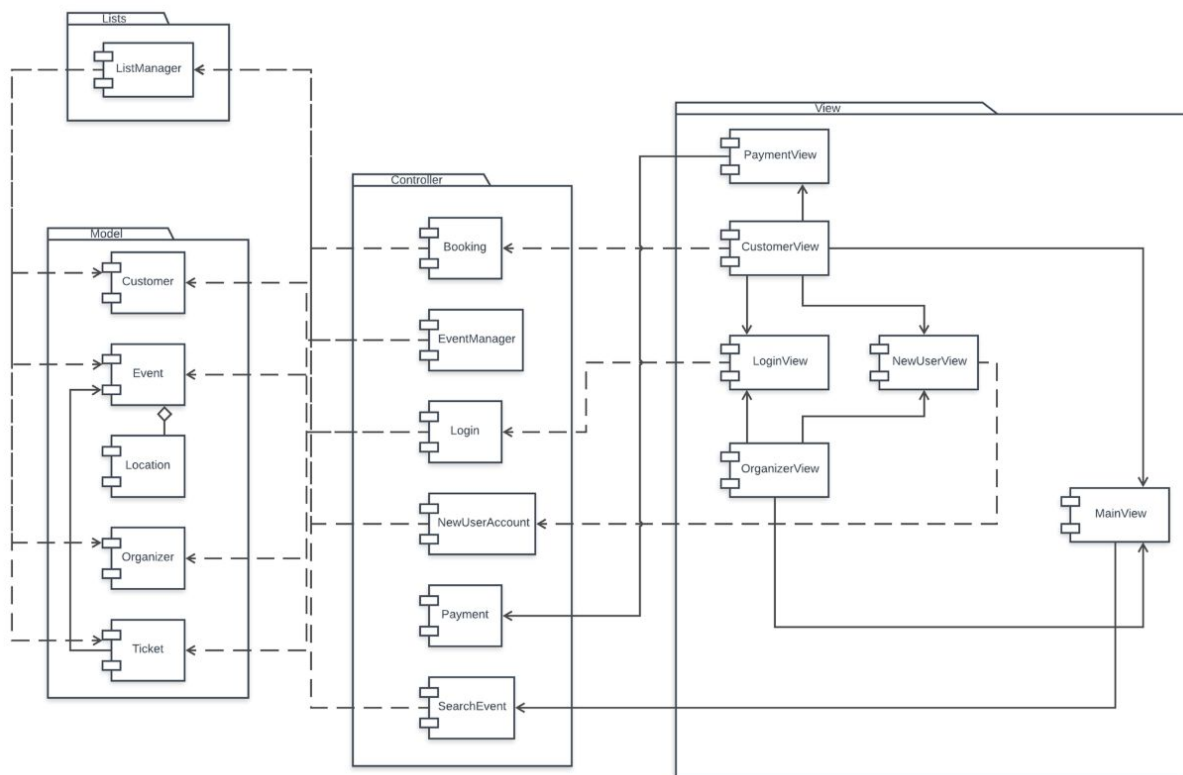
Alle klasser som ligger i View-pakken tar for seg kommunikasjon med brukeren av systemet. Her er det ikke noe avansert logikk, og den sender all informasjon som brukeren skriver inn videre til den rette klassen i Controller-pakken, som tar for seg behandling av informasjonen til brukeren, og gjennomfører all funksjonalitet utifra hvilke handlinger brukeren gjør. Ettersom systemet ikke bruker en database, lager systemet all data i minne i form av ArrayLists. All informasjon som får systemet til å fungere, blir lagret i listene som ligger i Lists-pakken.

Klassene i Controller-pakken tar for seg prosessering av data, så er den tett knyttet opp mot Model-pakken også. Klassene i Model-pakken inneholder bare såkalte "Blueprints" for hvordan et objekt skal bli laget, og hva det skal inneholde av data, samt metoder for å endre på disse objektene.

Nedenfor kan du se en enkel og overordnet modell på hvordan Model, View og Controller henger sammen i prototypen.



Nedenfor kan du se en modell for hvordan alle delene i prototypen henger sammen:



10.1 Prototypen og det endelige systemet

Prototypen er ment for å vise hvordan flyten i systemet kan være. Dette er verken en fasit eller en mal å følge, men den kan være nyttig for å vise hvordan systemet kan fungere. Prototypen tar for seg hovedfunksjonaliteten for dette systemet beskrevet i oppgaven, men prototypen mangler flere viktige elementer for at det skal i det hele tatt være mulig å ta i bruk systemet. Vi har satt et krav om at systemet skal være en Webapplikasjon, og i virkeligheten så er dette den mest optimale og realistiske måten å lage dette systemet på. Prototypen vi har laget er dog det stikk motsatte av en Webapplikasjon.

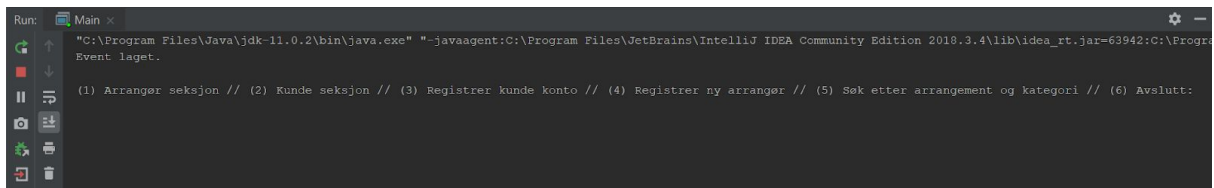
Men dette betyr ikke at prototypen nødvendigvis vil være til ingen nytte for de som skal bygge det endelige systemet. Koden som er skrevet i deler av prototypen, vil være mulig for utviklerne å ta i bruk som et utgangspunkt, og deretter modifisere deler av koden slik at det passer inn i det endelige systemet. Det å følge MVC-prinsippene i oppbyggingen av det endelige systemet, slik som vi har gjort i prototypen, kan være en god løsning for å dele opp logikken og gjøre systemet mer fleksibelt.

Det er verdt og merke at vi ikke setter det å følge MVC-prinsippene som et krav, eller valg av programmeringsspråk og rammeverk som et krav, ettersom firmaer har sine foretrukne måter å jobbe på, samt foretrukne programmeringsspråk og rammeverk. Å ha dette som noe krav for utviklingen av det endelige systemet synes vi derfor er unødvendig.

10.2 Skjermbilder fra prototype

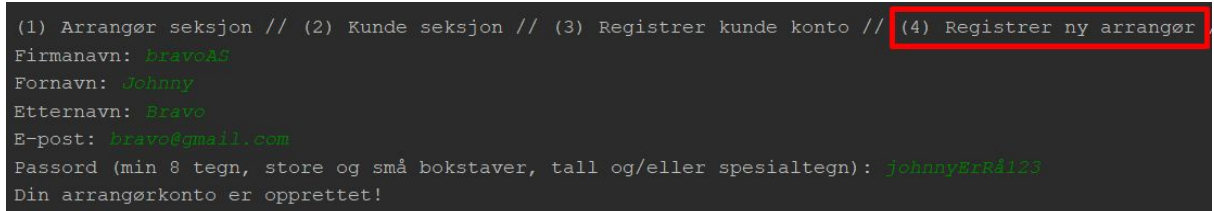
Under er det vedlagt skjermbilder med tilhørende forklaring til de funksjonalitetene gruppen har regnet som viktige for å visualisere hvordan prototypen fungerer. I samtlige skjermbilder er skriften systemet produserer presentert i fargen hvit - skriften som er presentert i grønt er det en bruker av systemet skriver inn.

Forside



Dette er den første siden i prototypen en bruker vil se. Her vil vedkommende kunne velge mellom flere ulike seksjoner. Om brukeren er en registrert kunde eller arrangør, kan de velge å gå til enten "arrangør seksjon" eller "kunde seksjon". For at brukeren skal få tilgang til de to forskjellige seksjonene må riktig innloggingsinformasjon være oppgitt. Hvis brukeren er helt ny i systemet kan de enten registrere seg som kunde eller arrangør i de respektive seksjonene "registrer kundekonto" eller "registrer ny arrangør". Det vil også være muligheter for en bruker og direkte søke opp arrangementer via denne forsiden.

Opprette arrangør konto



Slik vil det være for en bruker når han/hun skal opprette en arrangør konto i systemet. Systemet validerer i dette tilfellet om den oppgitte e-postadressen og passordet er gyldig. Hvis disse ikke hadde vært gyldig ville systemet bedt om at brukeren skriver inn informasjonen om igjen.

Arrangør seksjon

```
(1) Arrangør seksjon // (2) Kunde seksjon // (3) Registrer kunde konto // (4) Registrer ny arrangør // (5) Søk etter arrangement
Vennligst logg inn:
Epost: bravo@gmail.com
Passord: johnnyBrRd123

(1) Se mine arrangement // (2) Opprett arrangement // (3) Kontroller billett // (4) Marker billett som brukt // (5) Tilbake:
```

I dette tilfellet vil arrangører gå til "arrangør seksjonen". Det første vedkommende må gjøre er å logge inn med E-postadressen og passordet som han/hun registrerte seg i systemet med. Inne i selve arrangør seksjonen vil arrangøren bli presentert med en rekke valg han/hun kan utføre. Arrangøren kan velge:

- "Se mine arrangement"
- "Opprett arrangement"
- "Kontroller billett"
- "Marker billett som brukt"

Opprette arrangement

```
(1) Se mine arrangement // (2) Opprett arrangement // (3) Kontroller billett // (4) Marker billett som brukt //
Event navn: HiHaHo
Kategori: Ape-dans
Beskrivelse: Ell med å danse x, damer
Dato(YYYY-MM-DD): 2019-06-21
By: Fredrikstad
Gate: Apeveien 3
Aldersgrense: 20
Antall billetter: 100
Pris per billett: 210
Event laget.
```

Når en bruker skal opprette et arrangement må samtlige felter som blir vist over være fylt ut. Når alle felter er godkjent - får brukeren en beskjed nederst på skjermen i form av "Event laget" og arrangementet blir lagt til i sin respektive liste. Systemet lar arrangøren se en oversikt over sine arrangementer hvis han/hun trykker på "se mine arrangementer".

Søk etter arrangement og kategori

```
(1) Arrangør seksjon // (2) Kunde seksjon // (3) Registrer kunde konto // (4) Registrer ny arrangør // (5) Søk etter arrangement og kategori // (6) Avslutt:
(1) Søk etter arrangementnavn // (2) Søk etter kategori // (3) Tilbake til forsiden
```

Dette er en av funksjonalitetene som er vurdert som meget viktig for at kunden skal ha en så lettvinnt og positiv opplevelse med systemet. Når kunden trykker på "Søk etter arrangement og kategori", blir det ytterligere gitt kunden to valg nedenfor på skjermen. Kunden får da mulighet til og enten finne sitt ønsket arrangement via å søke etter navnet på arrangementet, eller å søke etter en kategori som er i kundens interesser. På denne måten blir det ikke en vanskelig jobb for kunden å finne sitt ideelle arrangement.

De to forskjellige søkene vil foregå på følgende måte som er vist under:

```

(1) Søk etter arrangementnavn // (2) Søk etter kategori // (3) Tilbake til forsiden
1
Søk etter arrangement:
sopptur
Følgende arrangment ble funnet:
Arrangement index: 1
Navn: Sopptur med Grete
Ledige billetter: 200

```

Systemet legger til rette slik at en kunde ikke trenger å søke etter det fulle navnet på arrangementer. De kan skrive inn deler av navnet - og fortsatt få presentert arrangementer som matcher søkeordet.

```

(1) Søk etter arrangementnavn // (2) Søk etter kategori // (3) Tilbake til forsiden
2
Søk etter arrangment i kategori:
friluft
Følgende arrangement ble funnet:
Arrangement index: 1
Navn: Sopptur med Grete
Ledige billetter: 200

```

Som vist i bildet over kan en kunde finne fram til sitt ønskede arrangement ved å søke etter kategorien på arrangementet. Denne kategorien er noe en arrangør legger ved som informasjon ved oppretting av arrangementet.

Kjøp av billetter

```

(1) Arrangør seksjon // (2) Kunde seksjon // (3) Registrer kunde konto // (4) Registrer ny arrangør /
Vennligst logg inn:
Epost: bruker@bruker.com
Passord: brukerABC123

(1) Se alle eventer // (2) Bestill billett // (3) Se mine billetter // (4) Tilbake: 2
===== Arrangement nummer 1 =====
Navn: Sopptur med Grete
kategori: Friluft
Beskrivelse: Ta med riktige klær og godt humør
Aldersgrense: 0
Ledige billetter: 200
Dato: 29.3.2019
Sted: Halden, BRA veien 6d
=====
Billett pris: 144.0

Velg event: 1
Velg antall billetter: 5
Total pris: 720.0,-

Skriv inn kort nummer(16 siffer): 1234567812345678
Utløpsdato(mm/åå): 10/19
CVC: 423
Booking godkjent!

```

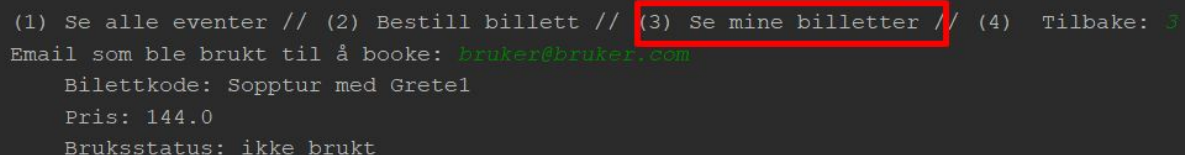
Når en kunde skal kjøpe billetter til et arrangement må de først logge inn med e-post og passord for å komme seg inn til kunde seksjonen. I kundeseksjonen velger han/hun om å 1) se en liste over alle arrangementer 2) bestille billett eller 3) se mine billetter. I dette tilfellet

velger kunden å bestille en billett. Kunden får da presentert de tilgjengelige arrangementene og velger ønsket arrangement ved å taste inn det ønskede "arrangement nummeret".

Etter kunden har valgt arrangement nummer, velger kunden antall billetter og får opp en totalpris som baserer seg på antall billetter a pris kroner x.

For at kunden skal kunne betale må betalingsinformasjonen være godkjent. Hvis alt er som det skal får kunden beskjed om at bookingen er godkjent og billettene legges til i kundens liste over billetter.

Se mine billetter



```
(1) Se alle eventer // (2) Bestill billett // (3) Se mine billetter // (4) Tilbake: 3
Email som ble brukt til å booke: bruker@bruker.com
Billettcode: Sopptur med Gretel
Pris: 144.0
Bruksstatus: ikke brukt
```

Som på bildet over kan en kunde gå inn å finne billetter som en tidligere har bestilt. Ved å skrive inn e-postadresse som ble brukt til bookingen vil alle billettene bli vist, samt bruks statusen og prisen på de.

11. Kjente svakheter og problemer i prototypen

De største kjente svakhetene i prototypen handler om såkalt "InputMismatch". Det dette betyr er at systemet i prototypen forventer å få inn en type input, f.eks. et tall, men brukeren skriver inn noe som ikke er et tall, som f.eks. en bokstav, et tegn o.l. I noen tilfeller vil prototypen krasje, og dette skjer stort sett når systemet spør om tall for navigering gjennom de forskjellige delene av systemet.

Andre svakheter i prototypen handler om muligheten til å gå ut av å gjennomføre ulike handlinger. Et konkret eksempel på dette er når man skal registrere en arrangør konto, så har man ikke mulighet til å gå tilbake når man vil i prosessen. For å gå ut av registreringen må man gjennomføre registreringen.

Lagring er også en kjent svakhet i prototypen, ettersom den ikke er koblet opp mot en respektiv database. Istedenfor at den er koblet til en database blir alt lagret i minnet. Det vil si at alt som blir registrert eller opprettet kun vil være tilgjengelig i minne så lenge programmet kjører og når programmet avsluttes vil alt blir "nullstilt".