UKE 10
Use case og kravhåndtering
UML modellering I

IN1030 - Gruppe 8

Plan for timen

Interessenter og aktører

UML og modellering

UML-diagrammer i IN1030 (i dag: Use Case diagram og sekvensdiagram)

Spørsmål?

Ukesoppgave 5 (5a, 5b, 5c & 5d)

Interessenter og aktører

Aktør: aktiv rolle, kommuniserer med systemet.

Interessent: kommuniserer ikke nødvendigvis med systemet.

Interessent

- En betegnelse for alle personer, grupper eller organer, som ***påvirkes av*** eller ***påvirker*** systemets utvikling/bruk. Både direkte og indirekte.
- En interessent kan være en person, gruppe, organisasjon eller institusjon.
- For en interessent har utforming av en løsning eller utfallet av en hendelse betydning.

Aktør

- En aktør er en samlebetegnelse for å angi alle grupper av personer som ***anvender*** systemet, eller øvrige systemer som ***blir anvendt*** av systemet.

- En aktør representerer en ***rolle*** som feks et menneske eller et annet system når det *kommuniserer* med dette systemet.

- En aktør ***kommuniserer*** eller ***interagerer*** med systemet, presentert via ett eller flere use case.

Primær- vs sekundæraktør

Primæraktør: eget **mål** i kommunikasjonen med systemet.

Sekundæraktør: trengs for at primæraktøren skal nå målet, kommuniserer også aktivt med systemet. Har ikke eget mål.

Interessent vs. aktør

- Interessent er et videre begrep enn aktør, det dekker alle som har interesse i systemet og som derfor vil legge føringer på systemet

- Skillet mellom aktør og interessent omhandler:
  - Rolle/tilknytning til bruker og utviklingen av systemet
  - Aktører må være enten
    - Brukere av systemet
    - Andre systemer som brukes av/bruker systemets kravspesifikasjon og utvikling

- Aktører er ofte interessenter, men ikke alle interessenter er aktører

Hvordan finner vi aktørene?

Spørsmål man kan stille for å finne aktører:

- Hvem skal bruke systemet?
- Hvem skal administrere systemet?
- Hvem
  - Tilbyr informasjon til systemet?
  - Bruker informasjonen fra systemet?
  - Fjerner informasjon fra systemet?
- Hvilke eksterne ressurser skal systemet bruke?
- Hvilke andre systemer skal kommunisere med dette systemet?

UML
(Unified Modeling Language)

Hva er UML?

Unified Modeling Language (UML) er en industristandard for datarelatert modellering, forvaltet av et internasjonalt konsortium kalt Object Management Group (OMG). (Wikipedia).

- Vi ønsker å vite hva skal lage før vi lager det.
- Vi må kommunisere hva vi skal lage med kunden.
- Vi ønsker å skape felles forståelse for arkitekturen, fordi vi samarbeider om koden.

UML - Hvorfor modellerer vi?

- I systemutvikling er modellering prosessen hvor man utvikler abstrakte modeller av et system .
- Det er viktig å forstå at én systemmodell ikke er en komplett representasjon av systemet, men at den kun viser ett perspektiv og ett ‘nivå’.
- ...derfor trenger man flere typer modeller som synliggjør ulike aspekter ved systemet; ulike modeller representerer ulike måter å se systemet på.

Når bruker vi UML?

	Når vi vil:
I kravanalysen	*..ha felles forståelse*
I designprosessen	*..beskrive tenkt system*
Etter implementasjon	*...dokumentere eksisterende system*

UML - Hvordan brukes modeller i systemutvikling?

1. **Som utgangspunkt for fokusert diskusjon om et eksisterende eller foreslått system**
   a. Omfang: → modellen må ikke være komplett, men bør dekke hovedpoengene i diskusjonen
   b. Notasjon → brukes uformelt

2. **Som dokumentasjon av et eksisterende system**
   a. Omfang → trenger ikke være komplett da man kan velge å dokumentere enkelte deler av systemet
   b. Notasjon → må være korrekt

3. **Som en detaljert systembeskrivelse som kan anvende som utgangspunkt for implementasjon av systemet**
   a. Omfang → modellene må være komplette
   b. Notasjon → må være korrekte

UML - Kravmodellering

Krav kan beskrives på mange måter:

- Tekst
- Strukturert tekst:
  - User story (brukerhistorie)
  - Use case (brukstilfelle)
- Modeller:
  - UML (Unified Modeling Language)
  - BPMN (Business Process Model and Notation)

Kravene må derfor være:

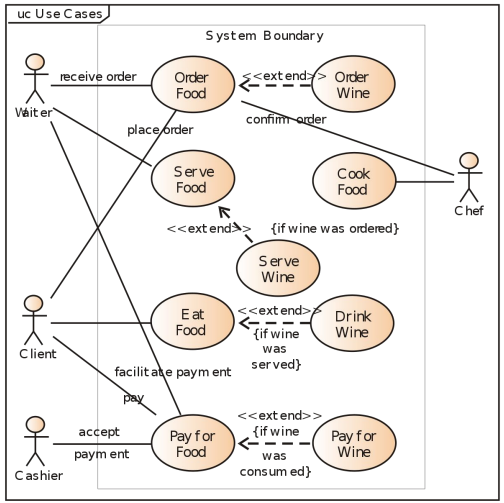
- **Forståelige** - alle interessenter/stakeholders må kunne forstå kravspesifikasjonen
- **Testbare** - vi må kunne avgjøre om det ferdige systemet gjør det det skal
- **Sporbare** - vi må vite hvilken del av koden som skal endres når det kommer nye krav

Hvilke 4 diagram skal dere kunne?

Hvilke diagram er dette?

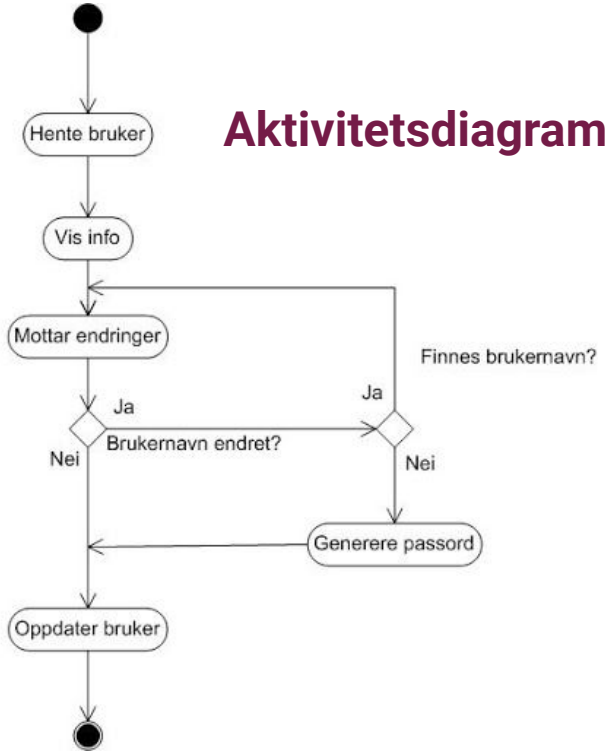
Use Case-diagram:

1.



2.

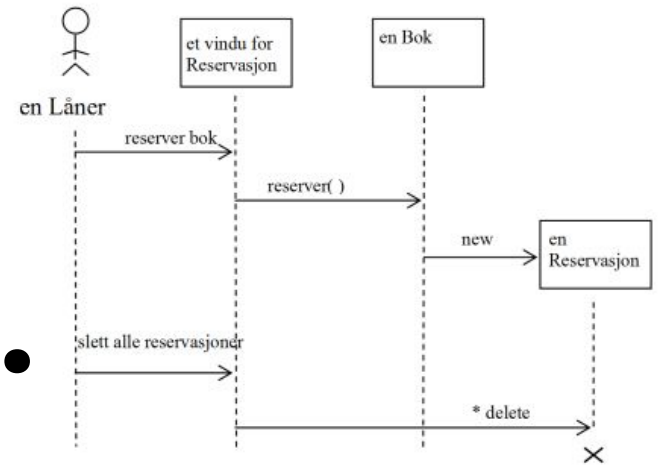
Aktivitetsdiagram:



Hvilke diagram er dette?

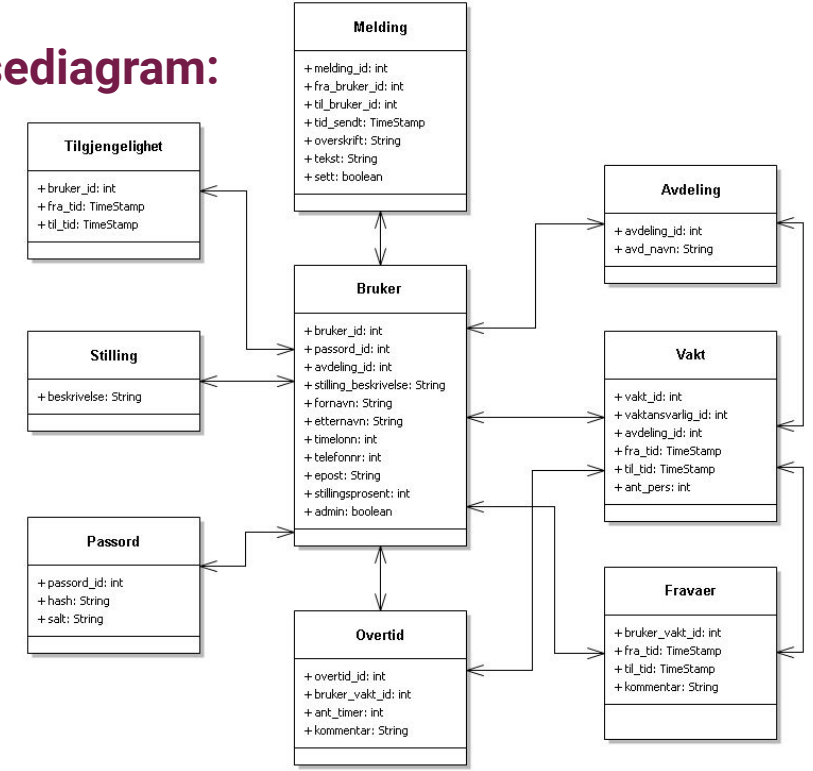
Sekvensdiagram:

3.

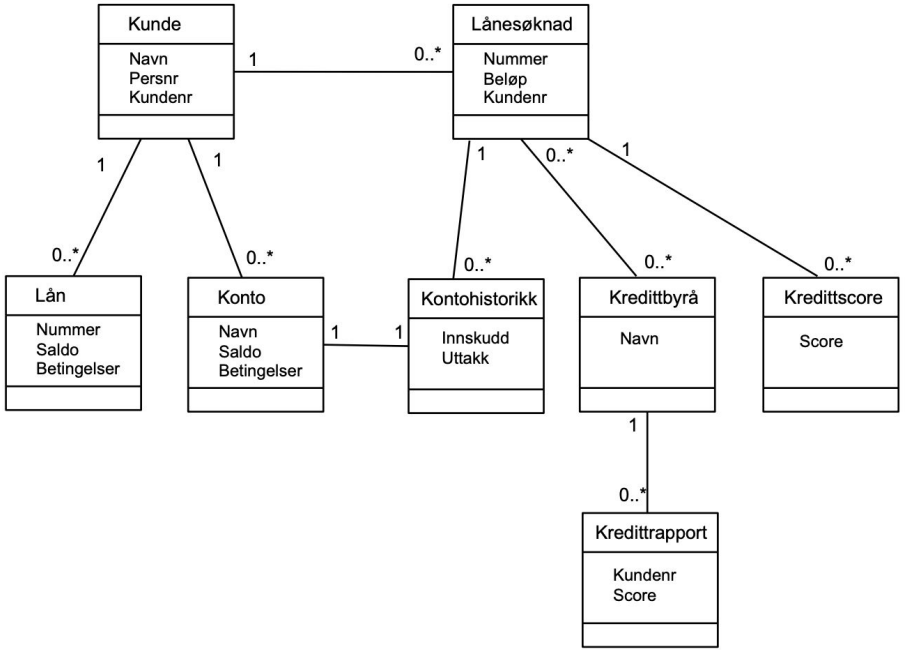


Klassediagram:

4.



Domenemodell - klassediagram uten metoder



Kort om “våre” modeller

Use case diagram: Viser interaksjon mellom et system og omgivelsene. Tar utgangspunkt i primæraktørs mål og hvordan sekundæraktører assisterer dette målet gjennom systemet.

Sekvensdiagram: Viser interaksjon og informasjonsflyten mellom aktørene og systemet og systemkomponentene i form av objektklasser.

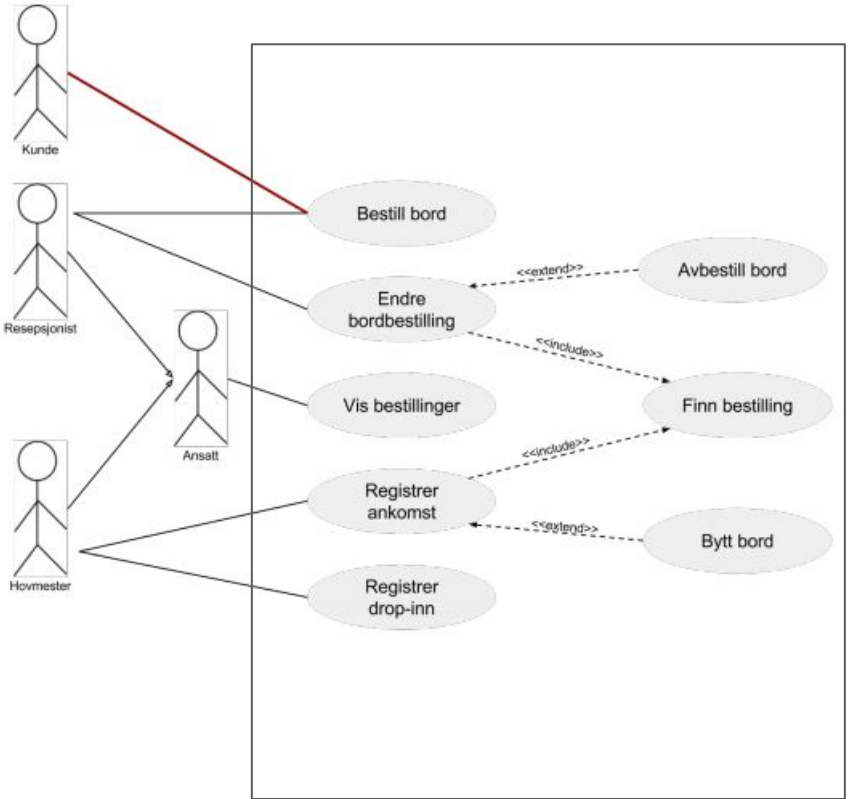
Klassediagram: Viser struktur: objektklasser av et system, deres attributter og metoder, og assosiasjonene mellom klassene.

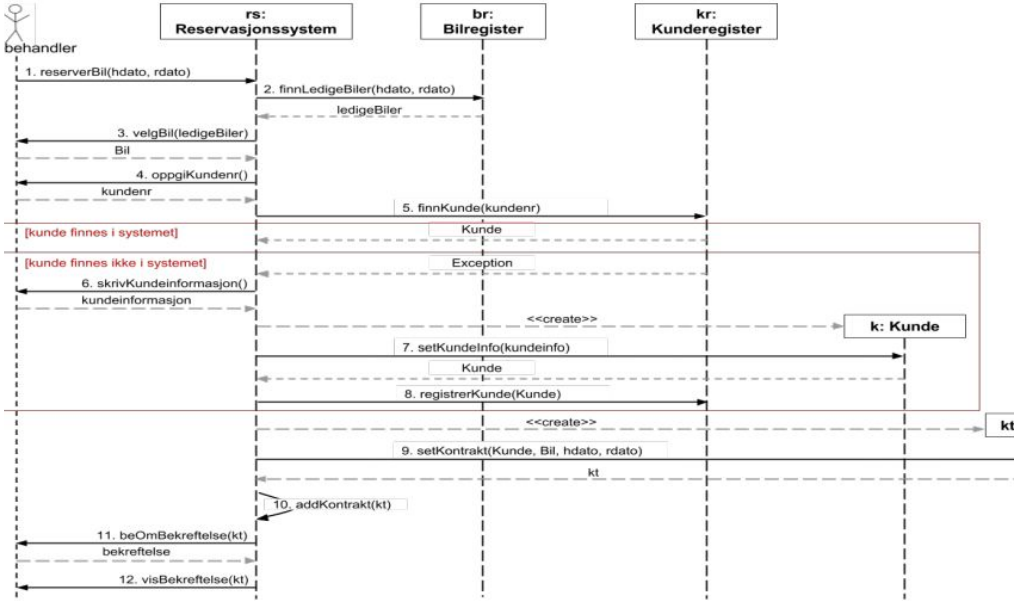
Aktivitetsdiagram: Viser aktivitetsflyten i en prosess eller dataprosessering.

OBS: Det er viktig at alle modeller for et og samme system samsvarer !
En metode som blir brukt i et sekvensdiagram må også være med i et klassediagram.

Interaksjonsmodeller

Use Case diagram & Sekvensdiagram





USE CASE-DIAGRAMMER

Use case I

- Vi snakker nå om use case (brukstilfelle), ikke user story (brukerhistorie)
- Handler om å identifisere aktørens mål → et use case
- Interaksjonsperspektivet: en beskrivelse av hvordan systemet oppnår et mål av verdi for en aktør

- Notasjon:
  - Figur for use case: oval
  - Merkelapp: navn på use case → verbfrase
  - I et use case diagram er aktøren strekfigurene, disse må ha et rollenavn
    - Primæraktører på venstre side, sekundæraktører på høyre.

Use case diagram II

- Viser systemets funksjonalitet og samspillet mellom systemet og omgivelsene (brukere, andre systemer, komponenter)

- Består av:
  - Aktører - tegnes som strekfigurer
  - Use case - tegnes som ovaler med “merkelapper”
    - Sorteres gjerne nedover i logisk rekkefølje.
  - Heltrukne streker mellom aktører og use case
  - Stiplet linje for include/extend mellom use case
  - Alle use cases som foregår i systemet, settes innenfor samme boks. Aktører og systemer udefra, plaseres ute (Scope).

User story / Brukerhistorie

- Korte setninger som briver rolle, funksjon og mål for systemet.
- Er knyttet til funksjonelle krav
- Kan knyttes til use case diagram

Eksempler:

Som kunde ønsker jeg å vite totalkostnaden på billeien.

Som kundebehandler ønsker jeg å se hvilke biler jeg kan leie ut til kunden.

Som kundebehandler ønsker jeg å reservere bil til kunden.

Hvordan finner vi use case?

Nyttige spørsmål for å finne use case:

- Hvilke ***mål*** ønsker aktøren å oppnå med bruk av systemet?
- Hvilke ***resultater*** vil aktøren oppnå med bruk av systemet?
- Hva er de ***viktigste oppgavene*** som aktøren ønsker at systemet *skal kunne utføre*?
- Vil aktøren **skape**, **lagre**, **endre**, **lese** eller **slette** data i systemet?
- Vil aktøren ha behov for å informere systemet om eksterne endringer?
- Har aktøren behov for å bli informert om hendelser i systemet?

Use Case diagram

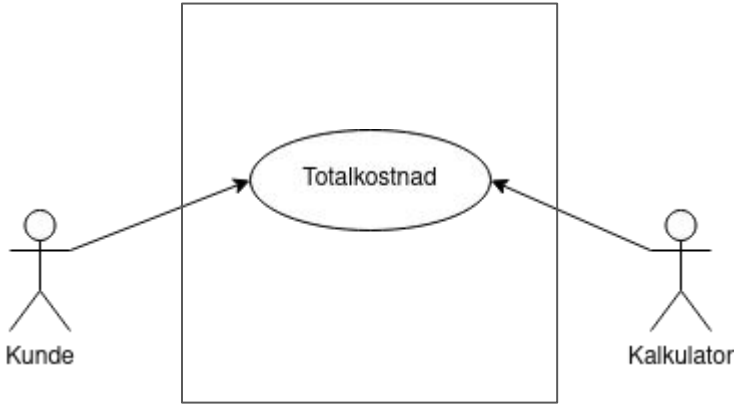
Hvilke mål har primæraktøren? → oval
Plasseres på venstre side.

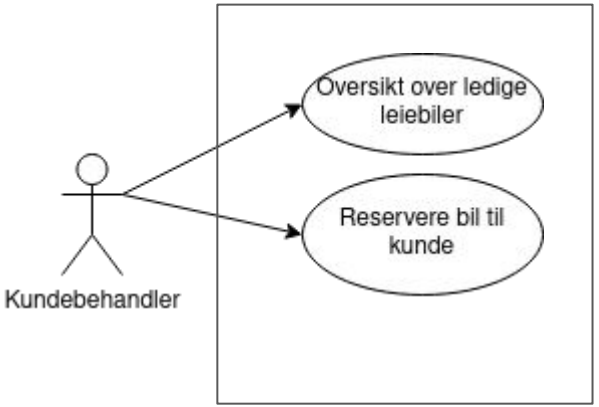
Hvem hjelper til med å nå primæraktørs mål? → Sekundæraktør
Plasseres på høyre side.

Fra brukerhistorie til Use Case diagram:
Som kunde ønsker jeg å vite totalkostnaden på billeien.

Som kundebehandler ønsker jeg å se hvilke biler jeg kan leie ut til kunden.

Som kundebehandler ønsker jeg å reservere bil til kunden.





Obs: ignorer pilene her - skal være bare streker

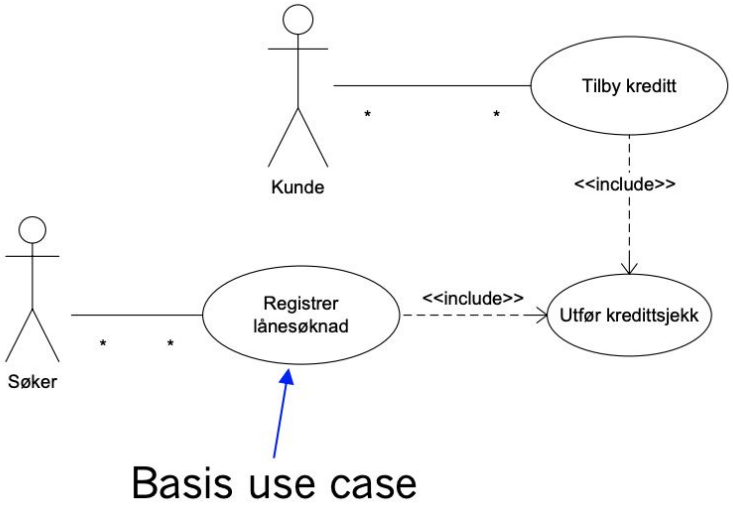
Use Case-relasjoner

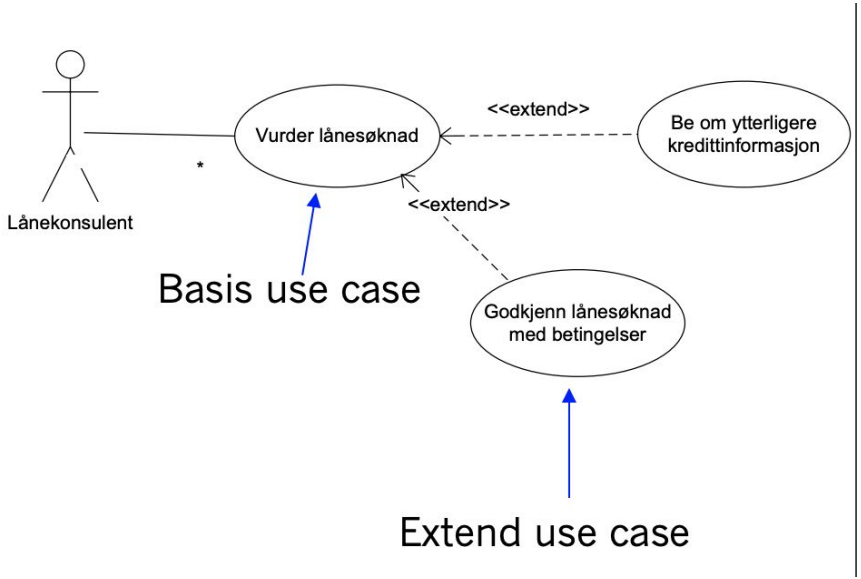
Include

- Et use case kan være en del av ett eller flere andre use case
- Indikerer at et (sub) use case inneholder nødvendig funksjonalitet for gjennomførelsen av et annet basiscase, og som skjer alltid for at basiscaset utføres.

Extend

- Et use case som beskriver tilleggsoppførsel som utføres under gitte omstendigheter
- Utvider oppførselen/funksjonalitet til et basiscase, som utføres under spesielle omstendigheter


Kunde
Tilby kreditt
<<include>>
Søker
Registrer lånesøknad
<<include>>
Utfør kredittsjekk
Basis use case


Lånekonsulent
Vurder lånesøknad
<<extend>>
Be om ytterligere kredittinformasjon
<<extend>>
Basis use case
Godkjenn lånesøknad med betingelser
Extend use case

Tekstlig beskrivelse av use case

Navn: Reserver bil
Primæraktør: Kundebehandler
Sekundæraktør: -
Prebetingelse: Ingen
Postbetingelse: Leiekontrakt for spesifisert bil og kunde med gitte utleiedatoer er opprettet

Hovedflyt:
1. Kundebehandler velger tidsintervall (hentedato og returdato)
2. Systemet returnerer en liste over tilgjengelige biler innenfor de spesifiserte datoene
3. Kundebehandler velger én av bilene.
4. Systemet ber om kundenr og finner kunden i systemet
5. Systemet bekrefter at bilen er reservert for den gitte perioden

Alternativ flyt punkt 2:
2.1: Det finnes ingen tilgjengelige biler i valgt tidsintervall.
2.2. Systemet opplyser om at det ikke er tilgjengelige biler innenfor oppgitt tidsintervall. 2.3. Kundebehandler oppgir et nytt tidsintervall (steg 1) eller avslutter bruksmønsteret.

Pre- og postbetingelser

Prebetingelser er betingelser eller tilstander som må være i systemet før et use case kan skje.

Postbetingelser er betingelser eller tilstander som må være tilstede eller oppfyldt før vi kan si at et use case er “ferdig”

Alternativ flyt

Hvis noe utenom det «vanlige» skjer, oppstår en alternativ flyt.

Dette fungerer på samme måte som i en tekslig beskrivelse.

Eks. Du skal logge inn, men har ikke bruker. Da må du inn i et **alternativt løp** for å opprette bruker. Deretter kan du begynne på starten igjen, på use caset logg inn.

SEKVENSDIAGRAMMER

Sekvensdiagram

Et **sekvensdiagram** viser sekvensene av interaksjon under et use case.

Notasjon:

- **Klasser og grensesnittet** i use caset representeres som bokser.
  Eks: Lege, Pasient og Legesystem.
- **Tidsflyten** går vertikaltnedover, slik som i sekvenstabell.
- **Aktøren** er represent som ***strekfigur*** på toppen av diagrammet med en stiplet linje vertikalet nedover (følger tidsflyten).
- **Interaksjon** mellom objektene er representert som ***piler.***
  - *Gjelder også når det er en request i systemet eller annet som ikke er reaksjon på andre beskjeder.*
- **Return message** er representert som ***stiplet linje,*** *eller beskjed tilbake fra recieving object/actor to sending object/actor.*
- **Navnet på pilen** indeketer *metodekall, parameter og returverdi*.
  Eks: FinnPasientInformasjon(fødselsnummer). Ønsket Pasientobjekt returneres.
- **Alternative flyt** skrives inn som ***bokse*** som viser de ulike alternative flyt.

Tips til modellering av sekvensdiagram (1)

1. Identifiser de ulike aktørene/objektene.
2. Lag et tenkt, tekstlig oppsett basert på hovedflyt.
3. Modeller steg for steg, basert på stegene i hovedflyten.
4. Inkluder alternativ flyt etter at du har laget en modell for hovedflyten.

Tekstlig beskrivelse av use case

Navn: Reserver bil
Primæraktør: Kundebehandler
Sekundæraktør: -
Prebetingelse: Ingen
Postbetingelse: Leiekontrakt for spesifisert bil og kunde med gitte utleiedatoer er opprettet

Hovedflyt:
1. Kundebehandler velger tidsintervall (hentedato og returdato)
2. Systemet returnerer en liste over tilgjengelige biler innenfor de spesifiserte datoene
3. Kundebehandler velger én av bilene.
4. Systemet ber om kundenr og finner kunden i systemet
5. Systemet bekrefter at bilen er reservert for den gitte perioden

Alternativ flyt punkt 2:
2.1: Det finnes ingen tilgjengelige biler i valgt tidsintervall.
2.2. Systemet opplyser om at det ikke er tilgjengelige biler innenfor oppgitt tidsintervall. 2.3. Kundebehandler oppgir et nytt tidsintervall (steg 1) eller avslutter bruksmønsteret.

Tips til modellering av sekvensdiagram (2)

1: Identifiser de ulike **aktørene** & **objektene**:

- Hvilket system er det snakk om? – *Reservasjonssystem for biler*
- Har vi eventuelle undersystemer? – *Bilregister/Kunderegister (avhengig av hvordan systemet er implementert)*
- Har vi eventuelle objekter? – *Kunde (objekter for de ulike kundene)/Kontrakt (objekt for utleiekontrakt)*
- Hvem skal interagere med systemet? – *Kundebehandler*

Aktør: **Kundebehandler.**
Klasser/Objekter: **Reservasjonssystem, Bilregister, Kunderegister, Kunde, Kontrakt**.

Tips til modellering av sekvensdiagram (3)

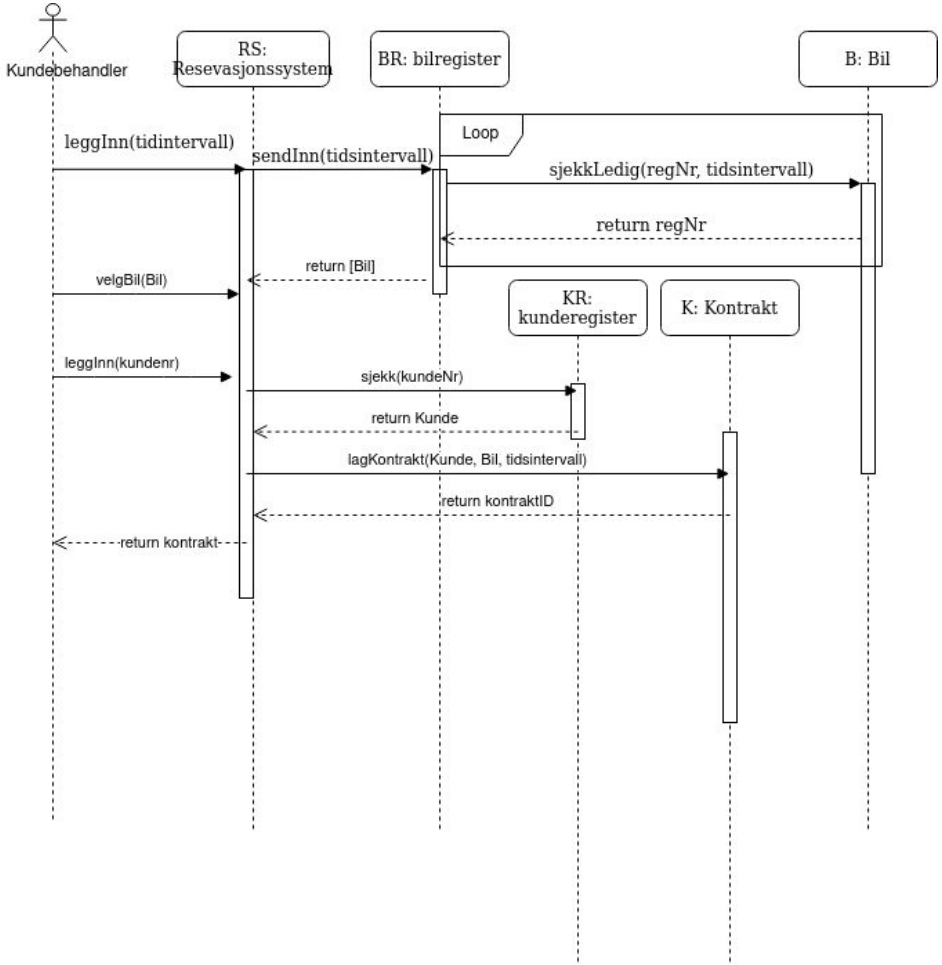
2: Lag et tenkt, tekstlig oppsett basert på hovedflyt:

- Aktør (Kundebehandler) til venstre.
- Interagerer med Reservasjonssystemet når hen velger tidsintervall.
- Da interagerer Reservasjonssystemet med Bilregister og henter ut tilgjengelige biler.
- «Viser» tilgjengelige biler til kunde.
- Kundebehandler (basert på «ekstern» interaksjon med kunde kanskje) velger bil og «sender» dette til Reservasjonssystem.
- Reservasjonssystem ber om kundenr.
- Kundebehandler gir kundenr.
- Reservasjonssystem finner kunde i Kunderegister.
- *Kundeobjekt på plass eller være med i Sekvensdiagrammet? Må det registeres som alternativ flyt?*
- Kontrakt blir laget med metode fra Kundeobjekt

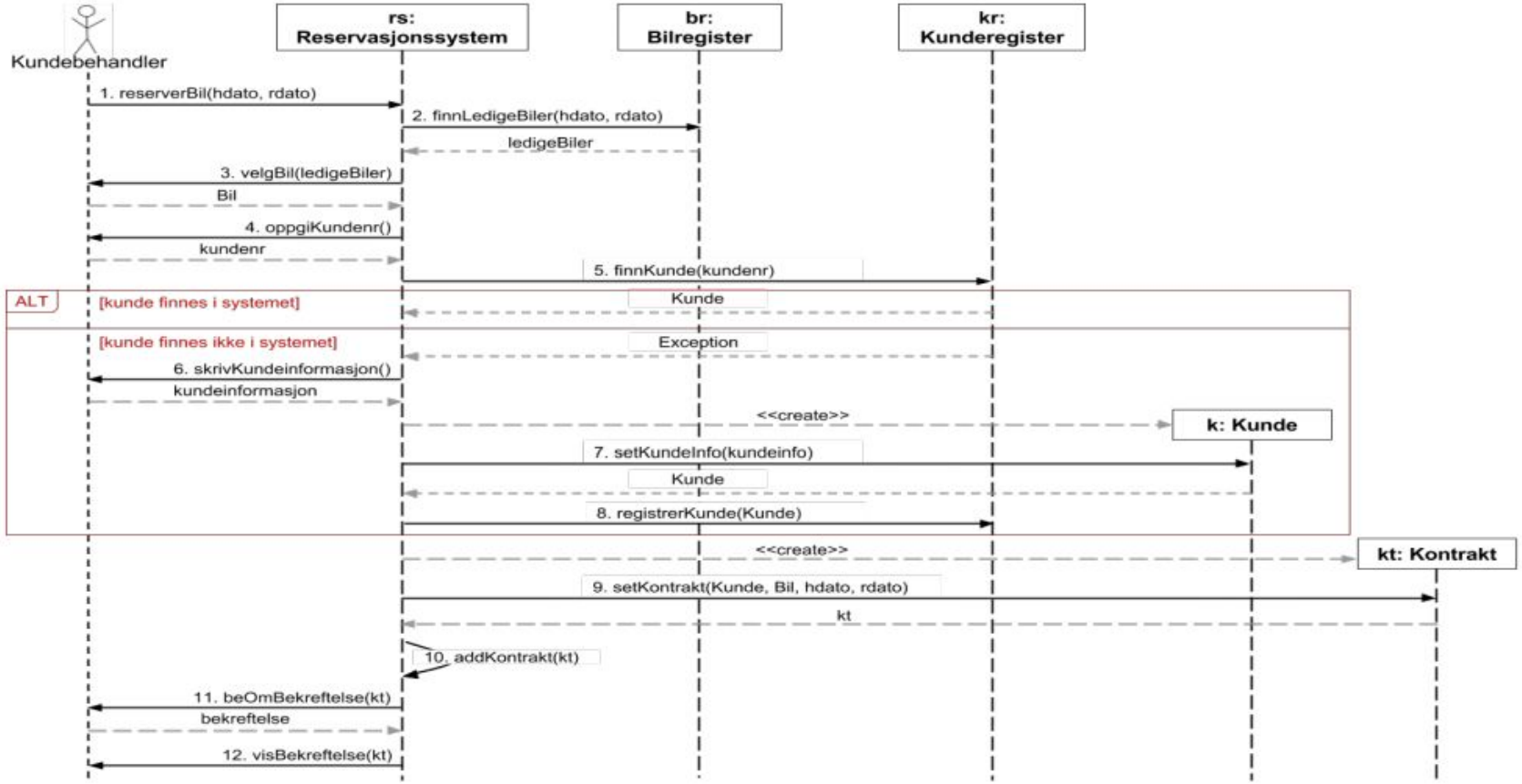
Sekvensdiagram

Hovedflyt:

1. Kundebehandler velger tidsintervall (hentedato og returdato)
2. Systemet returnerer en liste over tilgjengelige biler innenfor de spesifiserte datoene
3. Kundebehandler velger én av bilene.
4. Systemet ber om kundenr og finner kunden i systemet
5. Systemet bekrefter at bilen er reservert for den gitte perioden



Eksempel sekvensdiagram



Spørsmål?

Du kan stille spørsmål her eller på menti på:
https://www.menti.com/eo8obmgfb4 *eller kode:* ***5248 8163***

Oblig 4

Behov for gjennomgang?
Spørsmål rundt oppgavene?

Ukesoppgaver

(Med løsningsforslag for oppg. 5)

Systemet skal støtte bordreservasjoner og plassering i en restaurant. Kunder kontakter restauranten for å bestille / avbestille bord. En resepsjonist mottar samtalene. Bestillinger legges inn for et bestemt bord sammen med antall personer. For hver bestilling registreres kontaktperson med navn og telefonnummer.

Når gjester ankommer, blir de plassert ved sitt bord av hovmesteren, og bestillingen markeres med "ankommet". Hvis gjestene plasseres ved et annet bord, registreres dette bordbyttet. Tidspunktet et gitt bort må være ledig igjen kan også registreres. Kunder kan endre bestilling / avbestille på forhånd.

Det er mulig å få bord uten å ha bestilt på forhånd, gitt at ledige bord finnes. Når gjester får bord uten å ha bestilt dette, markeres tidspunkt, bord og antall i systemet, men ikke navn og telefonnummer.

Når nye bestillinger registreres i systemet, eller eksisterende bestillinger endres, skal skjermbildet umiddelbart oppdateres, slik at ansatte på restauranten alltid har oppdatert informasjon tilgjengelig.

SPØRSMÅL 5a

Spørsmål: Analyser beskrivelsen av et system som skal håndtere bord og bordbestillinger på en restaurant. **Finn aktører for systemet.**

SPØRSMÅL 5b

Spørsmål: Analyser beskrivelsen av et system som skal håndtere bord og bordbestillinger på en restaurant. **Finn use cases for systemet.**

SPØRSMÅL 5c

Spørsmål: Analyser beskrivelsen av et system som skal håndtere bord og bordbestillinger på en restaurant. **Lag et use case diagram for systemet.**

SPØRSMÅL 5a

Spørsmål: Analyser beskrivelsen av et system som skal håndtere bord og bordbestillinger på en restaurant. **Finn aktører for systemet.**

Svar:

- **Resepsjonist:** motta samtaler og håndtere bestillinger
- **Hovmester:** plassere gjester
- Videre kan man *generalisere* disse to til **“Ansatt”**, med samme mål: se oversikt over bestillinger/plasseringer.

SPØRSMÅL 5b

Spørsmål: Analyser beskrivelsen av et system som skal håndtere bord og bordbestillinger på en restaurant. **Finn use cases for systemet.**

Analyse:

Hva skal aktørene kunne gjøre? Hvilke resultater vil aktøren oppnå?

Identifiser de viktigste oppgavene (se etter verb)

- Skape / Lagre / Endre / Lese / Slette

Notasjon for use case-funksjoner:

Figur: Oval

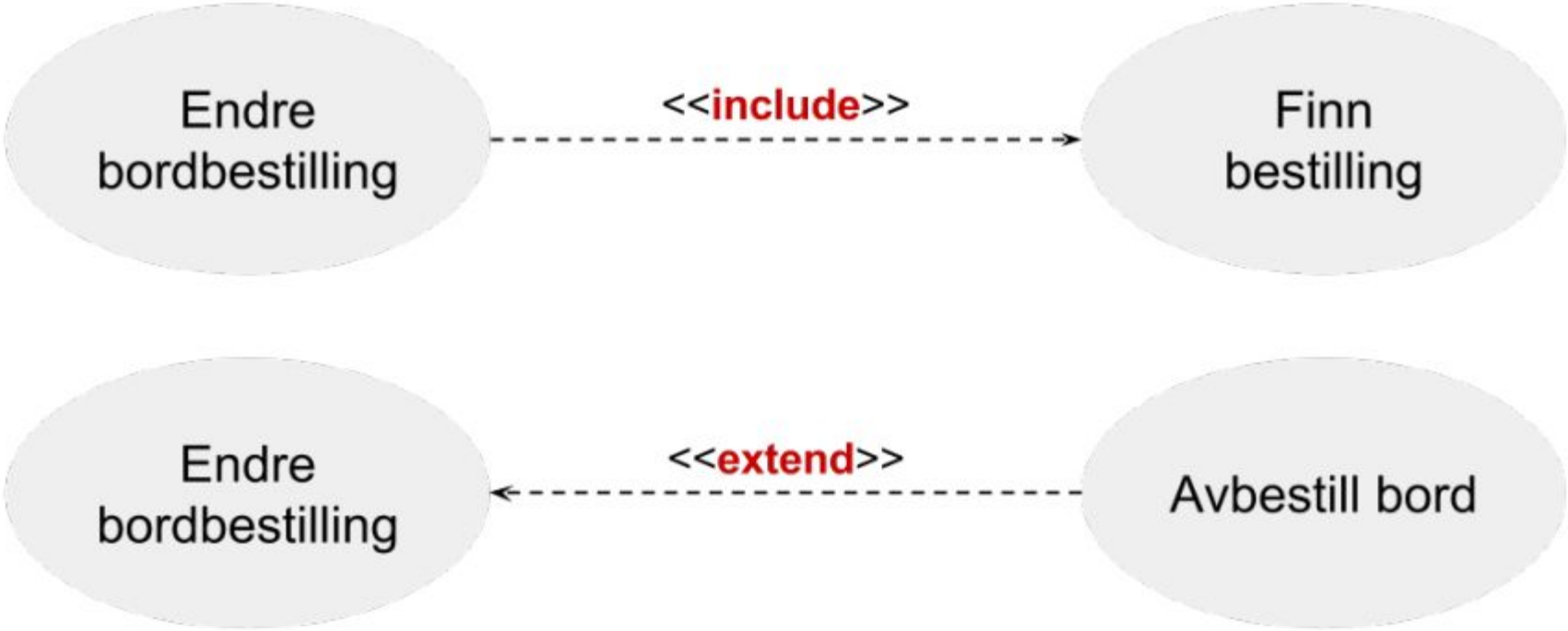
Merkelapp: Navn på use case → verbfrase

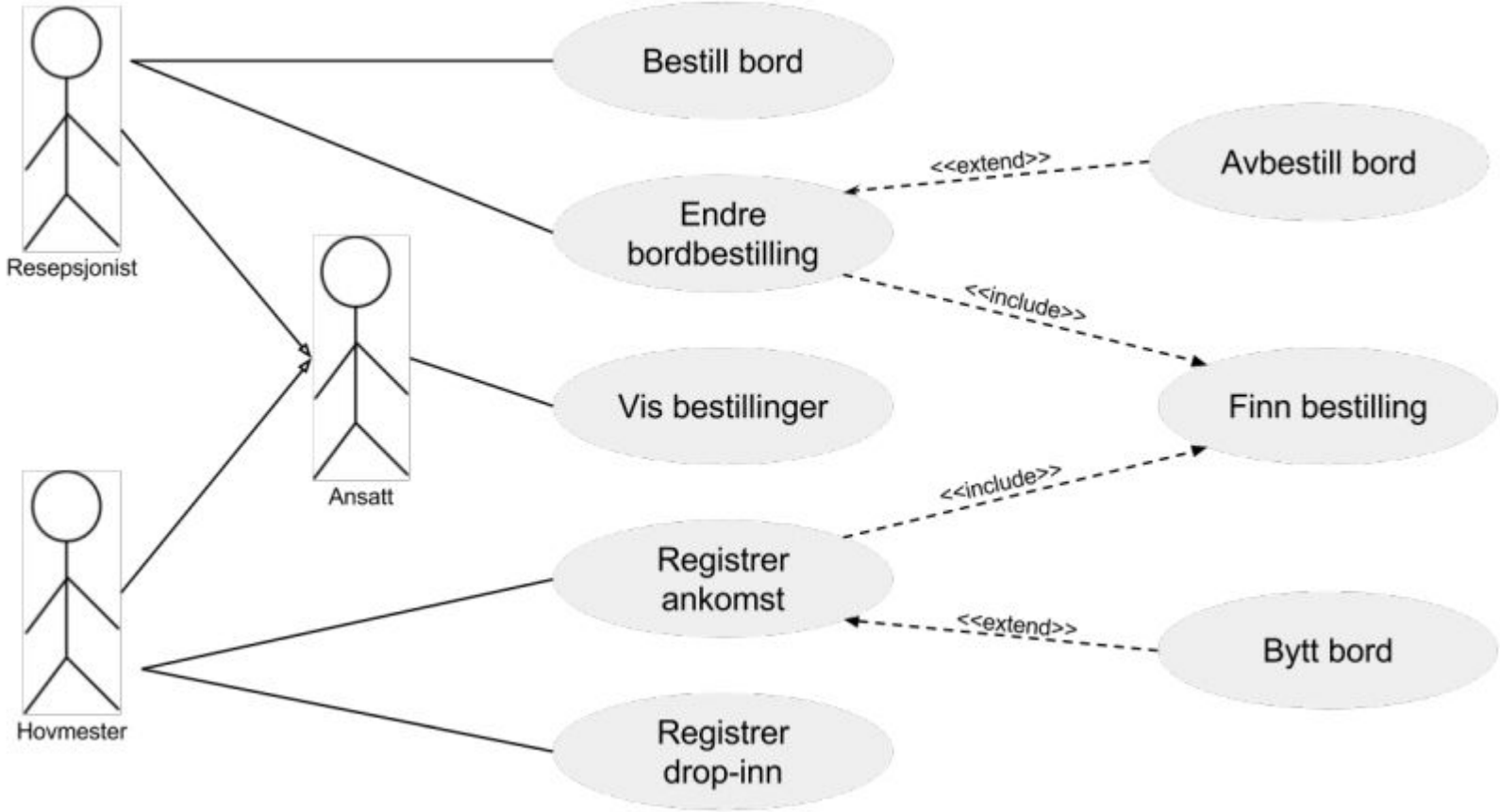
SPØRSMÅL 5c

Spørsmål: Analyser beskrivelsen av et system som skal håndtere bord og bordbestillinger på en restaurant. **Lag et use case diagram for systemet.**

Include-relasjonen: Indikerer at et (sub) use case inneholder nødvendig funksjonalitet for gjennomførelsen av et annet basiscase.

Extend-relasjonen: Utvider oppførselen / funksjonalitet til et basiscase, som utføres under spesielle omstendigheter.


Endre bordbestilling
<<include>>
Finn bestilling
Endre bordbestilling
<<extend>>
Avbestill bord


Resepsjonist
Ansatt
Hovmester
Bestill bord
Endre bordbestilling
Vis bestillinger
Registrer ankomst
Registrer drop-inn
Avbestill bord
Finn bestilling
Bytt bord
<<extend>>
<<include>>
<<include>>
<<extend>>

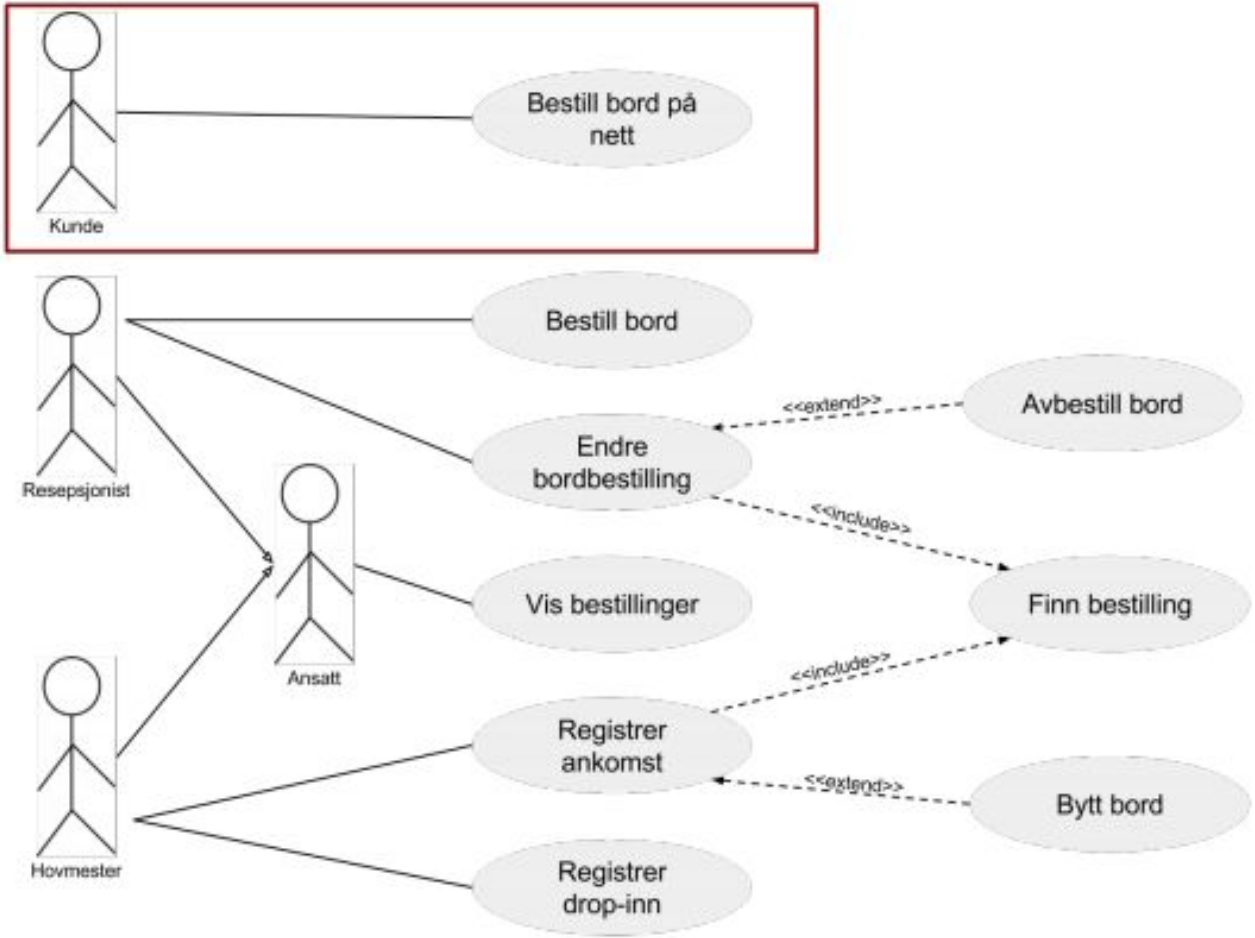
SPØRSMÅL 5d

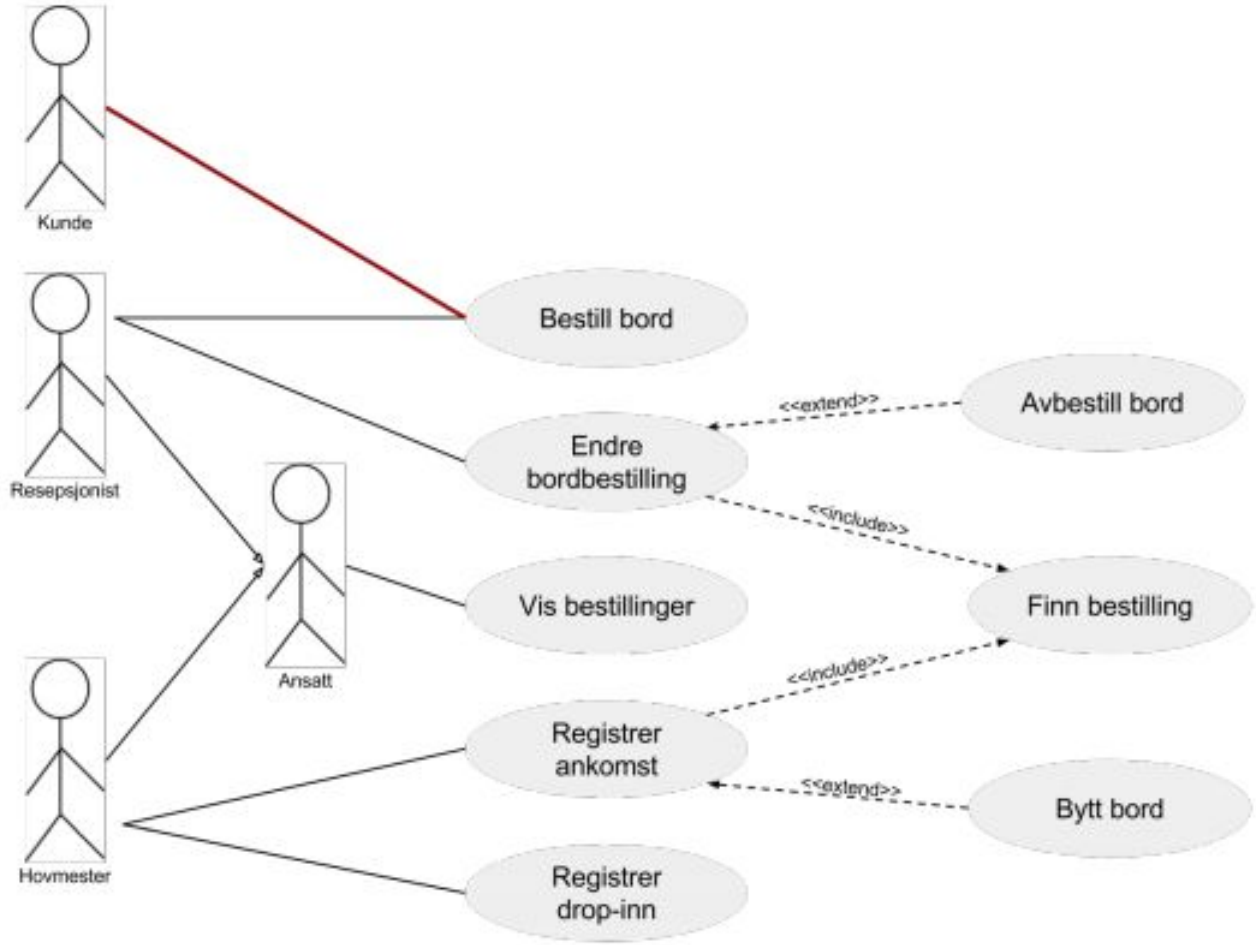
Spørsmål: Anta at en kunde kan bestille bord på nett i tillegg til over telefon. Gjør eventuelle modifikasjoner i løsningene i oppgave a, b og c.

Analyse:

Ny aktør **Kunde,** med mål: **Bestille bord på nett**

- Nytt use case?
  - Skal man opprette et nytt use case “Bestill bord på nett?”
  - Skal man beholde “Bestill bord”?
- Oppdater use case-diagrammet


Kunde
Bestill bord på nett
Resepsjonist
Ansatt
Hovmester
Bestill bord
Endre bordbestilling
Vis bestillinger
Registrer ankomst
Registrer drop-inn
<<extend>>
Avbestill bord
<<include>>
Finn bestilling
<<include>>
<<extend>>
Bytt bord


Kunde
Resepsjonist
Ansatt
Hovmester
Bestill bord
Endre bordbestilling
Avbestill bord
<<extend>>
<<include>>
Vis bestillinger
Finn bestilling
<<include>>
Registrer ankomst
<<extend>>
Bytt bord
Registrer drop-inn

Andre fine videoer:

Aktivitetsdiagram

Sekvensdiagram

Use Case-diagram

Denne uken

Modellering I: Use case og kravhåndtering

Neste uke

Modellering II: Objektorientert design

Frist: Oblig 4, 13/4 kl. 23.59

Takk for i dag!

Har du spørsmål, så send endelig en epost på nhmoller@uio.no