

# **UKE 14**

# **IT-Kontrakter + Repetisjon**

# **Modul B**

**IN1030 - Gruppe 9D / 2**

Oliver Ruste Jahren  
oliverrij@ifi.uio.no

# Program for i dag

Hva er kontrakter?

Hva er spesielt med IT-kontrakter?

Repetisjon - basert på Oblig 4

# Kontrakter

*“En kontrakt er en avtale som mellom partene etablerer en bindende forpliktelse til å gjøre eller til å unnlate å gjøre noe.”*

*Sitat fra forelæsningsopptak, i planen 5/5 2021*

*Tilbud + Aksept = Avtale*

*Ingen formkrav til de fleste typer kontrakter*

- *Muntlig / Skriftlig blant mange former*

*Kontraktens innhold*

- *Partene / Leveransen / Fremdriftsplan*
- *Bistandsforpliktelse / Spesifisert resultat / Definert tjenestenivå eller kvalitet*
- *Variabel pris / Fast pris / Måpris / Ytelsesbasert pris*

# Hva skriver man kontrakt på?



Resultat

Pris og/eller tidsestimat

Grad av fleksibilitet ifht  
produktbeskrivelsen

bla.

# Kontrakt-utfordring i IT-verdenen:



Kunde



Leverandør

# Kontraktsutfordring i IT-verdenen:

## Spesielt i IT-verdenen:

- Komplexitet
- Utvikling av noe nytt
- Sosiotekniske systemer: Skal bruke av mennesker og kan innebære endringer i arbeidsprosesser og organisering
- Abstrakte og usynlige systemer: i allefall for brukeren
- Mangel på modenhet i IT-bransjen.



Kunde



Leverandør

# Smidig → Flexibilitet i kontrakt

*"I gjennomføringen bestemmer du som kunde hva du skal ha" - Ståle L. Hagen*

# Plandrevet → Forutsigbarhet i kontrakt

Spesifisering av det vi skal ha før man går til leverandørmarkedet - man skal få det man spesifiserte at man ville ha.



# Den perfekte kontrakten ?

Fossefall: *vet hva vi skal ha og at det ikke er altfor omfattende.*

Seriefossefall: *ganske klart hva vi skal ha, begge parter evner å følge et rigid system for gjennomføring.*

Ressurskjøp: *vet mindre/ikke villig til å låse oss til hva slags produkt det skal skrives kontrakt for.*

Smidig **kontraktsmodell**: *ligner ressurskjøpsmodellen; leverandør er ikke ansvarlig for kontrakt, men for ressurser, gjennomføringsmodell og definerte ikke-funksjonelle krav.*

Hvilken mal?

# Norske standardkontrakter

Spesifisert resultat og fastpris

**SSA-T**

“Avtalen er egnet der leverandørens spesifiseringsarbeid (utarbeidelse av detaljspesifikasjon) ønskes gjennomført i nært samarbeid med kunden.”

[anskaffelser.no](https://anskaffelser.no)

Gradvis spesifisert resultat og pris

**PS2000**

**PS2000SOL**

Bistand betalt etter medgått tid

**SSA-B**

“Avtalen er egnet til konsulentkjøp når du har behov for kompetanse, men ikke vet hvordan sluttresultatet skal bli.”

[anskaffelser.no](https://anskaffelser.no)

# REPETISJON - Modul B

Etter å ha fullført IN1030:

- kan du drøfte samspillet mellom digital teknologi og individer, organisasjoner og samfunnet
- kan du utføre enkle brukerundersøkelser
- kjenner du til sentrale lover og forskrifter for utvikling av digitale systemer, og kan drøfte etiske problemstillinger
- kjenner du til ulike faser og aktiviteter som inngår i systemutvikling
- har du forståelse for samspillet mellom systemutvikling og ulike bruker og interessegrupper
- kan du anvende metoder og teknikker for kravhåndtering, utføre modellering ved hjelp av UML, og vurdere fordeler og ulemper ved forskjellige metoder og teknologier for systemutvikling

# NØKKELBEGREP

## Systemutvikling

### Prosessmodeller

Smidig utvikling

Plandrevet utvikling

Reell prosess

Scrum

Kanban

Fossefallsmodellen

### DevOps og kodehåndtering

Kravanalyse

Funksjonelle- og ikke-funksjonelle krav

Kontrakt

Kostnad

### Prosjektplanlegging

UML modellering

Leverandør

Kravhåndtering

Implementering

Kunde

Risikoanalyse

Trusselmodellering

Testing

Produkteier

Vedlikehold

# Hvilke hovedaktiviteter inngår i en systemutviklingsprosess?

Planlegging

Kravinnsamling

Kravanalyse

Design

Programmering

Testing

Konfigurasjonsstyring

Versjonshåndtering

hva som skal lages og innenfor hvilke rammer/krav.

design og programmering.

validerer at systemet er det kunden vil ha.

modifiseres etter kunden og markedets krav/behov.

**Hvordan velge riktig  
prosessmodell?**

# Hva slags system skal bygges?

System av det annet system

Individuelle applikasjoner

Interaktive transaksjons-baserte applikasjoner

→ vurder: hva har vi allerede, hvilken kontekst skal vi inn i, hvem er kunden/bruker og hvilken interesse har kunden/bruker?

Spørsmål: Forskjellen på interessent og aktør?



# Hvordan påvirker aktører og interessenter valg av systemutviklingsprosess?

**Aktør:** noen (kan og være et annet system) som bruker systemet aktivt. Enten: ha eget mål med å anvende systemet (primæraktør), Eller: hjelper primæraktøren med å oppnå sitt mål i systemet (sekundæraktør).

**Interessent:** blir påvirket eller påvirker systemets utvikling og/eller drift.

## Plandrevet:

- Tydelig mål fra aktører
- Gjerne få interessenter og aktører
- Vanskelig å opprettholde kommunikasjon med aktør.

## Smidig:

- Tvetydige mål
- Gjerne mange aktører og interessenter
- Effektiv kommunikasjon med aktører

MEN OBSOBS: Kunde har alltid "final say" ! Og det finnes gjerne en kontrakt.

# Smidig vs Plandrevet: oppgave: 3 hovedforskjeller

## Smidig:

- Dynamisk samarbeid med kunde/produkteier
- Dynamisk reviderbar kravspesifikasjon
- Prioriterer å håndtere kravendring sammen med kunde
- Inkrementell levering av produktets funksjoner

## Plandreven:

- Må holde seg til planen til en utgave av ferdig produkt er levert (evt endringshåndtering mulig, men tidkrevende)
- Fokus på dokumentering av prosess
- Lengre og mer detaljert for- og kravanalyse.
- Statisk kravspesifikasjon
- Prioriterer å utvikle systemet basert på forhåndsbestemt plan
- Oftest kun ett endelig produkt

# Hvordan bestemme seg for Smidig eller Plandrevet?

Oftest: vil kravspesifikasjonen endre seg?

Ja: smidig. Nei: plandrevet.

Også: er det et stort system som krever masse ressurser (penger, tid, utviklere) og forutsigbarhet?

Ja: plandrevet. Nei: smidig.

# Hvordan bestemme seg for Scrum eller Kanban?

Vurder følgende:

1. Trenger vi Daily-Standups?
2. Trenger vi retrospektiv?
3. Trenger vi struktur?
4. Er det tydelig hvilke oppgaver som får oss frem til målet?
5. Klarer vi å estimere hvor lang tid oppgavene tar?
6. Trenger vi å kunne gjøre endringer *nårsomhelst* i utviklingsprosessen?
7. Trenger kunde en garantert inkrementell levering av produktet?

Hvis svaret er ja:

*Hvilke taler for Scrum?*

*Hvilke taler for Kanban?*

Hvis svaret er ja:

Scrum: 1., 2., 3., 5., 7.

Kanban: 4., 6.

# Lage kravspesifikasjon - hvordan?

**Mål:** både funksjonelle og ikke-funksjonelle krav. Gjerne også måloppnåelse og testing av kravene :))

Brukerhistorier - basert på identifiserte interessenter → identifiserer primær og sekundæraktører og funksjonelle krav (i hovedsak).

Kravspesifikasjonen kan ha enten veldig naturlig språk, eller veldig unaturlig. Bør følge et system.

Funksjonelle krav:

Hva skal systemet gjøre? Hvilke mål skal det tilfredstille?

Ikke-funksjonelle krav:

**Produktkrav:** hvordan skal produktet fungere?

**Organisatoriske krav:** hvordan skal systemutviklingsprosessen gjennomføres?

**Eksterne krav:** hvilke krav har eksterne omgivelser (konteksten)?

Skal være mål/testbare - presise!

## Eksempler:

**Funksjonelt krav:** Systemet skal generere oversikt over mest brukte kinoer.

**Ikke-funksjonelle krav:**

**Produktkrav:** Nettsiden skal håndtere opptil 5000 samtidige brukere.

**Organisatoriske krav:** Systemutviklingen skal holde et budsjett på maks 30 millioner NOK.

**Eksterne krav:** Systemets betalingsløsning må følge NF-kinos krav til samarbeidspartnere (se vedlegg 3).

# Eksempel med WCAG

## Funksjonelle krav:

- Siden/nettstedet skal vise innhold

## Ikke-funksjonelle krav:

- *Hvordan skal innholdet vises?* Her kommer eksempelvis WCAG-prinsippene inn:

At innhold skal vises etter disse retningslinjer og med disse formål

# Eksempler tester:

Stresstesting

Brukertesting

Sjekklister

Direkte målbart (ja/nei)

Ekspertisehjelp



# UML-modellering

Forstå systemet → systemutviklingsprosess mer effektiv

# Oversikt over diagrammer

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

Henger sammen med tekstlig beskrivelse.

**Brukes når:** Man ønsker å se interaksjonen mellom aktør og system.

**Sekvensdiagram:** Viser interaksjon og informasjonsflyten mellom aktørene og systemet og systemkomponentene i form av objektklasser. Et kodenært diagram (eks. bruker metodekald) (mer detaljert enn use case diagrammene)

Henger sammen med tekstlig beskrivelse og klassediagram.

**Brukes når:** Man ønsker å se på interaksjonen på et mer systemnært nivå.

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

(**Domenemodell:** Klassediagram uten metoder)

**Brukes når:** Man ønsker å få en oversikt ved å vise alle klasser, metoder, assosiasjoner osv. i systemet, og vise arkitekturen.

**Aktivitetsdiagram:** Viser aktivitetsflyten i en prosess eller dataprosessering. Grafisk representasjon av hendelsesflyten i et use case.

**Brukes når:** Man ønsker å vise flowet, prosesser eller omgivelser for et system.

**Tekstlig beskrivelse:** En tekstlig beskrivelse av en use case tar for seg interaksjonen mellom systemet og bruker, ved en nummerert liste som beskriver hvert interaksjonssteg for seg.

**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.**

# USE CASE DIAGRAM

Må med: aktør og use case

Er sekundæraktør med? OBS: kun hjelpe.

Interessenter er *ikke* aktør

System er *ikke* aktør

Ikke-funksjonelle krav er *ikke* use case.

**Sammenheng med funksjonelle krav !**

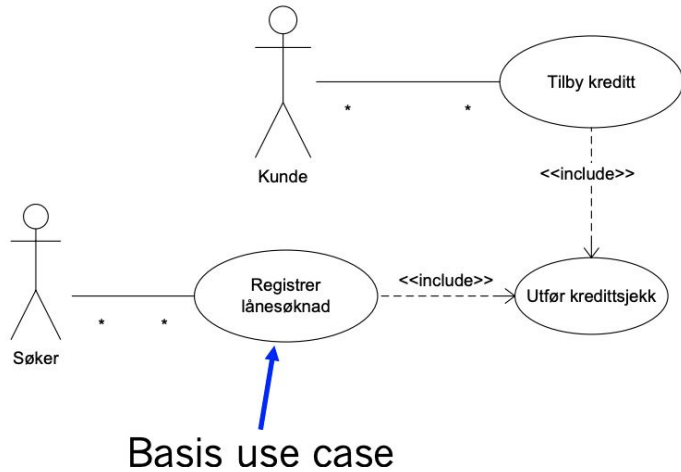
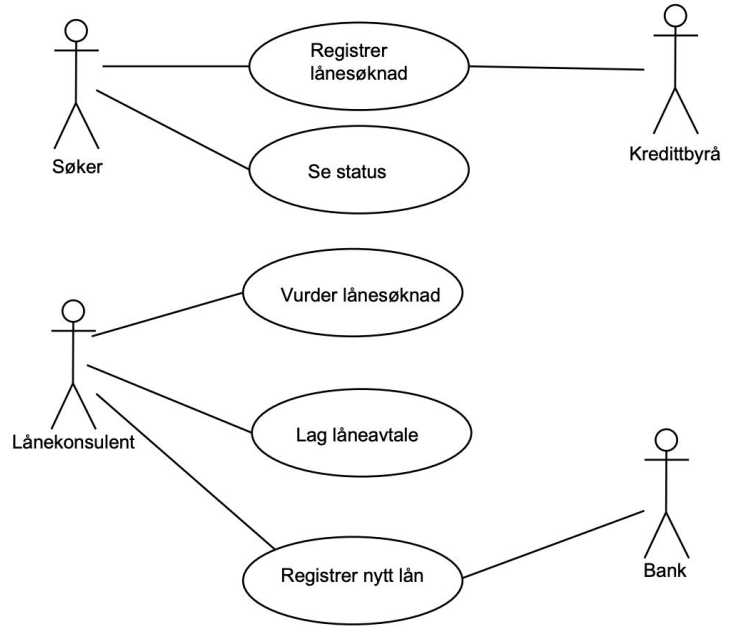
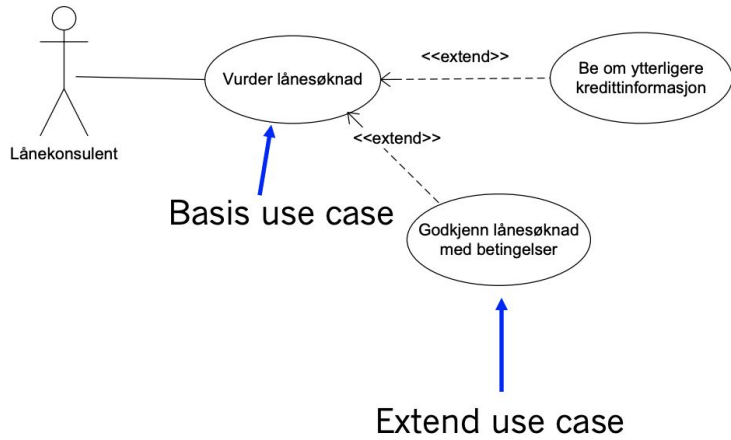
MULTIPLE CHOICE:

Hva innebærer include-relasjonen?

- a. At man inkluderer flest mulig primærbrukere i et use case og viser relasjonen mellom disse
- b. At man inkluderer sekundærbrukere i et use case og viser relasjonen mellom primær og sekundærbrukere
- c. Et use case som kan være en del av ett eller flere andre use case
- d. Et use case som beskriver tilleggsoppførsel som utføres under gitte omstendigheter

Hva innebærer extend-relasjonen?

- e. At man inkluderer flest mulig primærbrukere i et use case og viser relasjonen mellom disse
- f. At man inkluderer sekundærbrukere i et use case og viser relasjonen mellom primær og sekundærbrukere
- g. Et use case som kan være en del av ett eller flere andre use case
- h. Et use case som beskriver tilleggsoppførsel som utføres under gitte omstendigheter



# Tekstlig beskrivelse: Hva må med?

Navn:

Primæraktør:

Sekundæraktør:

Prebetingelser:

Postbetingelser:

Hovedflyt:

Alternativ flyt:

# Eksempel

**Navn:** Registrere ny bruker via nettside

**Primæraktør:** Bruker

**Sekundæraktør:** Ingen

**Prebetingelser:** Ingen

**Postbetingelser:** Ny bruker opprettet i systemet

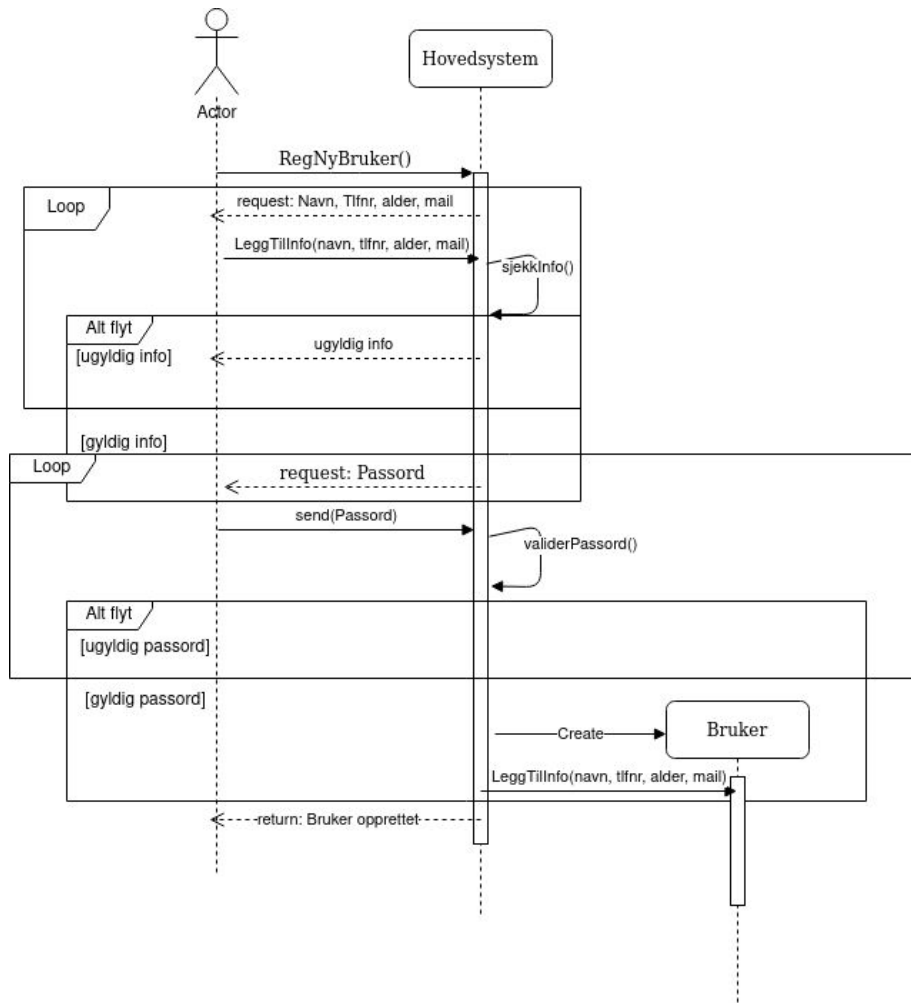
## Hovedflyt:

1. Bruker velger “registrer ny bruker”
2. Systemet ber bruker om å oppgi navn, tlfnr, alder og mailadresse
3. Bruker skriver inn informasjon
4. Systemet validerer informasjonen
5. Systemet ber bruker om å opprette passord
6. Bruker skriver inn valgt passord
7. Systemet validerer passord
8. Systemet legger inn bruker i systemet
9. Systemet sender bekreftelse på skjerm og mail

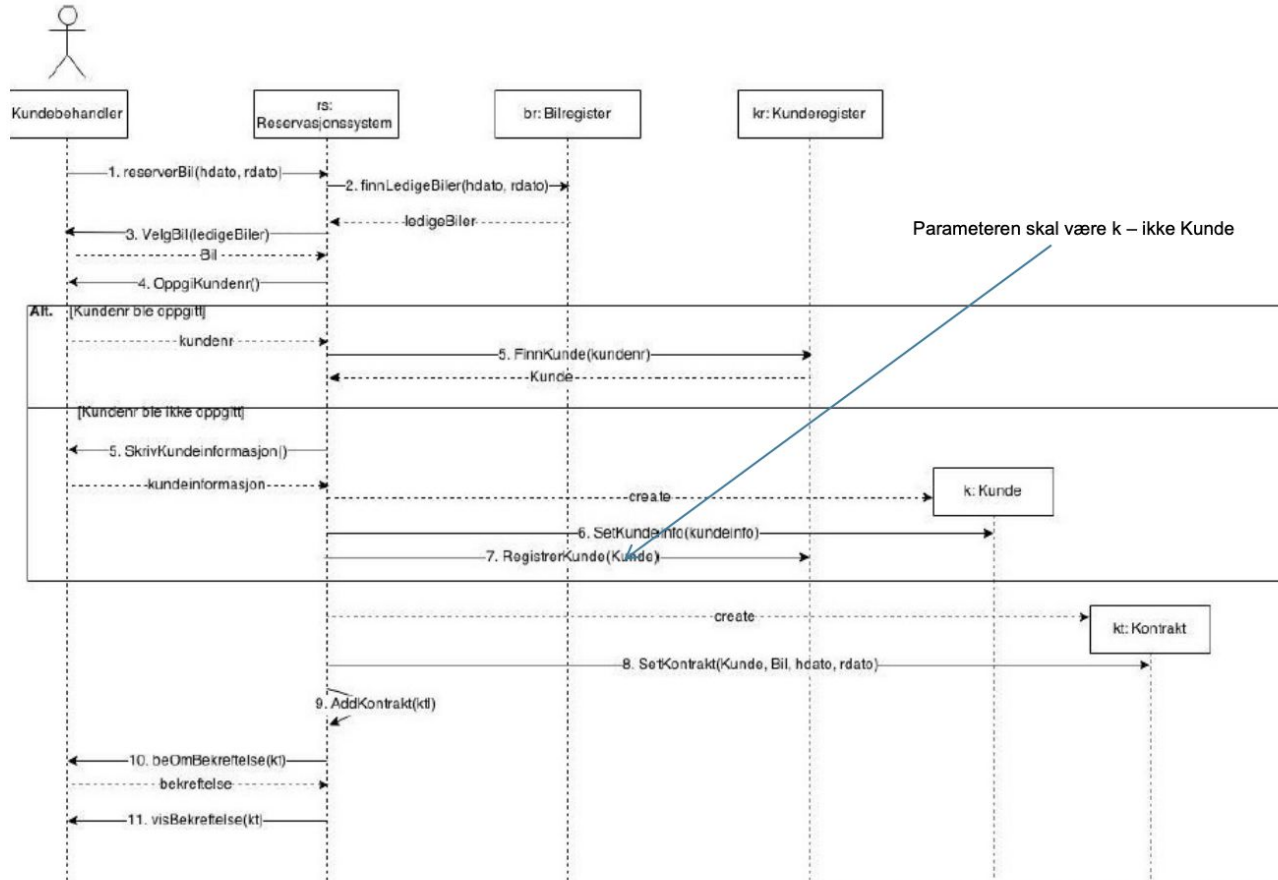
## Alternativ flyt:

- 4.1. Ugyldig tlfnr, for få siffer
- 4.2. Returnerer tilbake til steg 2

- 7.1. Ugyldig passord, mangler vanskelighetsgrad
- 7.2. Systemet ber bruker taste inn et annet passord
- 7.3. Returnerer til steg 6



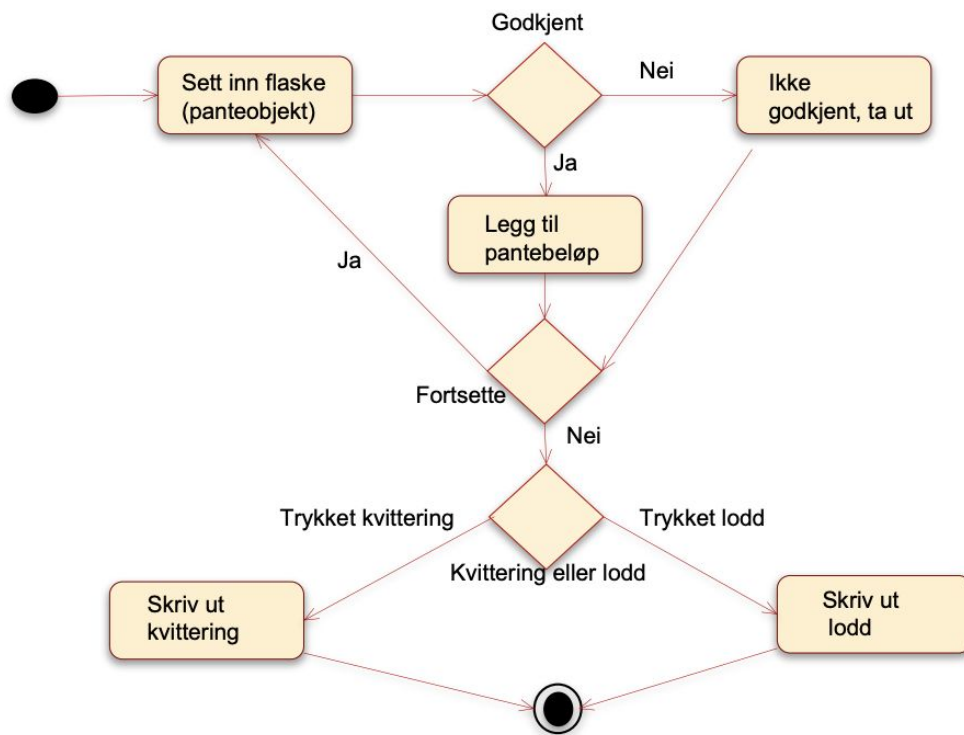
# Sekvensdiagram - Reserver bil



EMILIE HALLGREN OG KRISTIN BRÆNDEN

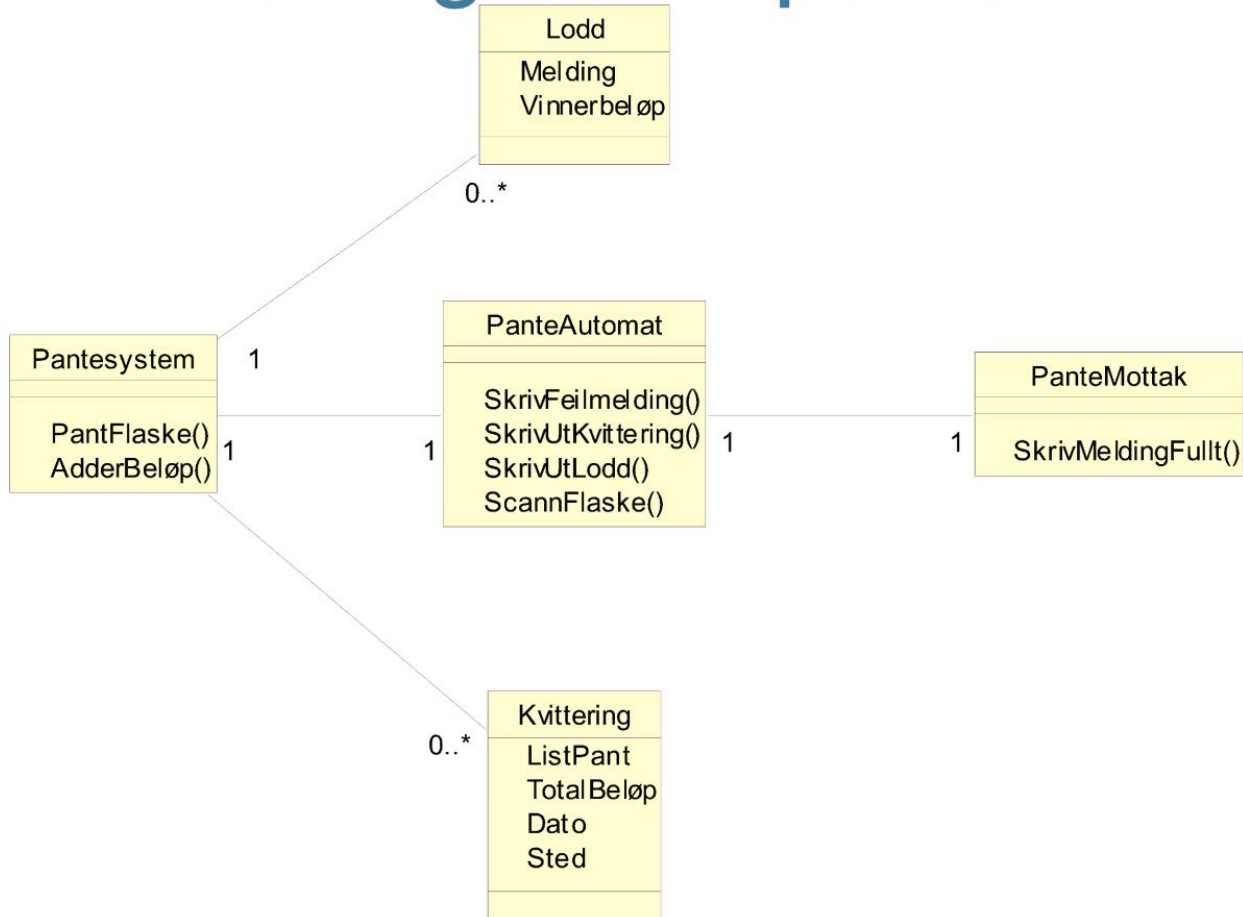


# Aktivitetsdiagram – pante flasker



Screen shot fra forelæsning 14/4

# Klassediagram – pante flasker



Spørsmål?

**Tidligere eksamensoppgaver kan finnes på  
semestersiden**

**[Link til oversikt over tidligere eksamensoppgaver](#)**

**oliverrij@ifi.uio.no**