Hospitalsprojekt

**2012**

Anette Stidsing, Niklas Renner, Thomas Michael og Martin Hana

14-12-2012



Indholdsfortegnelse

[Afgrænsning 4](#_Toc343245137)

[Systemudvikling 5](#_Toc343245138)

[1. Systemdefinition 5](#_Toc343245139)

[2. Problemområde 5](#_Toc343245140)

[2.1. Klasser 6](#_Toc343245141)

[2.2 Hændelser 7](#_Toc343245142)

[3. Anvendelsesområde 7](#_Toc343245143)

[3.1. Aktørtabel 7](#_Toc343245144)

[Aktører 7](#_Toc343245145)

[3.2. Brugsmønstre 8](#_Toc343245146)

[Opret patient 8](#_Toc343245147)

[Udskriv patient 8](#_Toc343245148)

[3.3. Funktionsliste 9](#_Toc343245149)

[3.4. Brugergrænseflade 9](#_Toc343245150)

[Database 11](#_Toc343245151)

[1. Entiteter 11](#_Toc343245152)

[2. Forbindelser ER-diagram 12](#_Toc343245153)

[3. Konvertering 13](#_Toc343245154)

[3.1 konverteringsalgoritmen 13](#_Toc343245155)

[3.2 EER diagram 14](#_Toc343245156)

[4. Data 14](#_Toc343245157)

[5. Test 15](#_Toc343245158)

[-- Test af foreign key 15](#_Toc343245159)

[-- delete, insert, update, select 15](#_Toc343245160)

[-- Udtræk af data fra mere end en tabel 15](#_Toc343245161)

[Programmering 16](#_Toc343245162)

[1. Hvordan har vi valgt at løse opgaven 16](#_Toc343245163)

[1.1 Cardlayot i modsætning til tabbed pane 16](#_Toc343245164)

[1.2 GUI og brugervenlighed 16](#_Toc343245165)

[1.3 UML diagram 17](#_Toc343245166)

[2. Klasser og centrale metoder 17](#_Toc343245167)

[2.1 Database klassen 17](#_Toc343245168)

[2.2 Query klassen 18](#_Toc343245169)

[2.3 Main klassen 19](#_Toc343245170)

[3. Exception handling 19](#_Toc343245171)

[4. Test 21](#_Toc343245172)

[Oversigt Test Cases 21](#_Toc343245173)

[4.1 Test 1 - Indlæg to patienter i samme seng 21](#_Toc343245174)

[4.2 Test 2 - Opret patient 23](#_Toc343245175)

[4.3 Test at man kun kan indlægge en patient i en seng som lægeteamet er tilknyttet 24](#_Toc343245176)

[IT i organisationen 27](#_Toc343245177)

[1. Organisationsmodel, der beskriver ansvar myndighed, arbejdsdeling og redegør for de funktioner der skal være. 27](#_Toc343245178)

[2. Hvordan sikrer ledelsen sig at IT-sikkerheden overholdes i forbindelse med patientjournalen? 27](#_Toc343245179)

[3. I forbindelse med anskaffelsen af det nye system bedes I redegøre for hvilken måde ledelsen bør implementerer systemet for de ansatte, så alle bakker op om det nye system. 29](#_Toc343245180)

[4. Redegør hvilke fordele der kan være for patienterne og de ansatte ved indførelsen af patientjournalen 29](#_Toc343245181)

[5. Vurder konsekvenserne ved at bruge det nye system, dels økonomisk men også organisatorisk. 29](#_Toc343245182)

[Ansvarsområder 31](#_Toc343245183)

[Systemudvikling 31](#_Toc343245184)

[Database 31](#_Toc343245185)

[Programmering 31](#_Toc343245186)

[IT i organisation 31](#_Toc343245187)

# Afgrænsning

Systemet som ønskes udviklet skal kunne håndtere oprettelse af patienter op Næstved hospital. Derudover Skal man kunne indlægge patienterne og vælge en ledig seng samt udskrive patienten igen. Det er ikke muligt at oprette læger, lægeteams eller nye senge i systemet.

Design dokumentet i systemudviklings afsnittet blev ikke lavet. Dette skyldes manglende viden og planlægning i gruppen.

# Systemudvikling

## 1. Systemdefinition

Der udvikles et delsystem til brug i Næstved hospital, til daglig administration af patienter og indlæggelser. Systemet skal kunne håndtere oprettelse, indlæggelse og udskrivning af patienter samt søgning på patienter. Man skal i systemet kunne oprette en patient samt søge på en patient ved at indtaste patientens CPR nummer. Hvis patienten er indlagt, skal man kunne se hvornår patienten er blevet indlagt, i hvilken afdeling, stue og seng patienten befinder sig i og hvilket lægeteam patienten er tilknyttet. Desuden skal det være muligt at udskrive patienten. Ved indlæggelse af en patient, skal man kunne tildele patienten en ledig seng, samt tilknytte patienten et lægeteam.

## 2. Problemområde

### 2.1. Klasser

#### Patient tilstandsdiagram

### C:\Users\Anette Stidsing\Dropbox\1.SemesterOpgave\Systemudvilkling\Tilstand for patient.jpeg

Tilstandsdiagrammen fra klassen patient. Her vises hvis en patient ikke er oprettet i systemet, så bliver han det før han kan indlægges.

#### Indlæggelse tilstandsdiagram

### C:\Users\Anette Stidsing\Dropbox\1.SemesterOpgave\Systemudvilkling\Tilstand for indlæggelse.jpeg

Tilstandsdiagrammet fra klassen indlæggelse. Her vises at når patienten er indlagt på en afdeling og skal flyttes til en anden afdeling, så bliver patienten udskrevet og indskrevet til den afdeling patienten flyttes over til. Dvs. at når objektet indlæggelse ændres, så nedlægges det og oprettes en ny.

### 2.2 Hændelser

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Hændelser/klasser | Patient | Indlæggelse | Lægeteam | Læge | Afdeling | Afsnit | Værelse | Seng |
| Opret Patient | + |  |  |  |  |  |  |  |
| Søg på patient | \* | \* | \* |  |  |  |  | \* |
| Patient indlægges | \* | + | + |  |  |  |  | + |
| Patient udskrives |  | + |  |  |  |  |  |  |
| Patient død | + |  |  |  |  |  |  |  |

Når vi har lavet listerne over kandidater til klasser og hændelser, vurdere vi dem systematiskved hjælp af generelle og specifikke vurderingskriterier. Som et led i denne evaluering knytter vi klasser og hændelser sammen i en hændelsestabel, der fastholder resultatet af klasseaktiviteten.

## 3. Anvendelsesområde

### 3.1. Aktørtabel

|  |  |  |
| --- | --- | --- |
| Brugsmønstre/Aktør | Læge | Sygeplejerske |
| Opret Patient |  |  |
| Søg på patient |  |  |
| Indlæg patient |  |  |
| Udskriv patient |  |  |

### 

### Aktører

***Oversigt:***  Vi har identificeret to aktører – læge og sygeplejerske - og ialt 4 brugsmønstre. Nedenfor beskrives rollerne for systemets to aktører

***Læge***

**Formål:** En læge skal rådgive patienten om behandlingen og give oplysninger om brug og dosering af medicin, risici m.v.

**Karakteristik:** En læge er en person, der er godkendt lægevidenskabeligt fakultet på et universitet, og som har aflagt lægeløftet.

***Sygeplejerske***

**Formål:** En sygeplejerske har som regel ansvaret for én gruppe patienter ad gangen. I tæt samarbejde med læger. En vigtig del af dette arbejde er blandt andet at indsamle de nødvendige oplysninger, så patienten kan få den rette behandling og pleje.

**Karakteristik:** En sygeplejerske udfører, leder, formidler og udvikler sygepleje. Sygeplejens kerneområde er omsorgen til de syge. Dette opnås primært gennem bevidst brug af de mellemmenneskelige relationer.

### 3.2. Brugsmønstre

### Opret patient

Brugsmønster: Oprette en patient igangsættes af en ansat. Den ansatte indtaster CPR nummer i systemet. Da CPR nummeret ikke eksisterer, vil der når man trykker på knappen søg, synliggøres en knap "opret patient". Derefter anmodes den ansatte om at indtaste patientens fornavn, efternavn, postdistrikt, adresse samt kunden telefonnummer. Når den ansatte har gjort dette, vælges seng og det lægeteam som patienten skal tilknyttes.

Objekter: patient, lægeteam, indlæggelse, seng.

Funktioner: Opret Patient (Opdatering)

### Udskriv patient

Brugsmønster: Når en patient skal udskrives, gøres dette ligeledes af en ansat. Den ansatte søger på patientens CPR nummer, og vælger funktionen "udskriv".

Objekter: Patient, Indlæggelse

Funktioner: Udskriv Patient (Opdatering)

### 3.3. Funktionsliste

|  |  |  |
| --- | --- | --- |
| Funktioner | Kompleksitet | Type |
| Søg Patient | Kompleks | Aflæsning |
| Opret Patient | Medium | Opdatering |
| Indskriv Patient | Medium | Opdatering |
| Udskriv Patient | Simple | Opdatering |
|  |  |  |

Ud fra vores systemdefinition og analyse af vores brugsmønster har vi fastlagt funktionerne der detaljere komplekse funktioner, hvorefter vi systematisk har vurderet funktionslisten og dens specifikke kompleksitet og funktionstyper.

### 

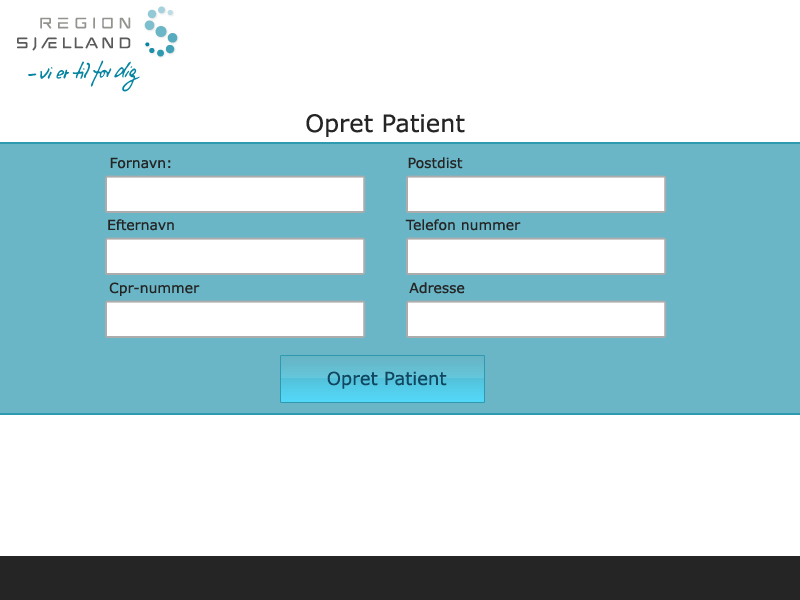
### 3.4. BrugergrænsefladeC:\Users\Anette Stidsing\Dropbox\1.SemesterOpgave\Systemudvilkling\Forside.jpg

Grænseflader forbinder system til alle relevante aktører i omgivelserne. Det understreges i følgende situation:

**Grænseflade:** *Faciliteter der gør et systems model og funktioner tilgængelige for aktører.*

Vi har ud fra vores klassediagram, brugsmønster og funktionsliste udarbejdet og fastlagt grænsefladens elementer.

Når man indtaster et cpr nummer, vil programmet skifte til patientens status, og hvis patienten ikke er oprettet vil den skifte til siden, hvor der oprettes patienter.

Siden Opret Patient, er sat op så der kun oprettes patient og når man opretter, bliver man ført videre til Patient Info, hvor man indlægger patienten.

Opret Patient, har vi valgt at den skal bestå af fornavn, efternavn, postdistrikt, adresse, tlf-nummer, cpr-nummer og en knap når patient endeligt oprettes.

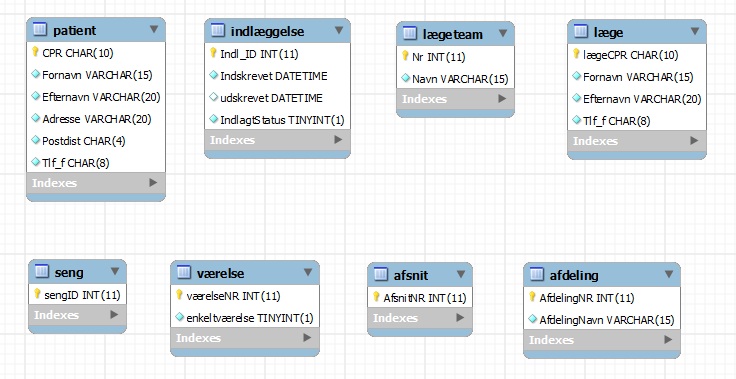


Siden Patient Info viser patienten og hvor patienten er indlagt; afdeling, afsnit, værelse og lægeteam. På siden er der to knapper, den ene ”Indlæg” som indlægger patienten, og den anden ”Udskriv” udskriver patienten.

# Database

## 1. Entiteter

Databasen skal indeholde oplysninger om patienter, indlæggelser og den seng som patienten ligger i. Samtidig skal patienten under indlæggelse være tilknyttet et lægeteam, bestående af fire læger. Derfor oprettes følgende entiteter:



I entiteten patient anvendes CPR som primærnøgle.

I entiteten indlæggelse anvendes indl\_ID som primærnøgle.

I entiteten lægeteam anvendes Nr som primærnøgle.

I entiteten læge anvendes lægeCPR som primærnøgle.

i entiteten seng anvendes sengID som primærnøgle.

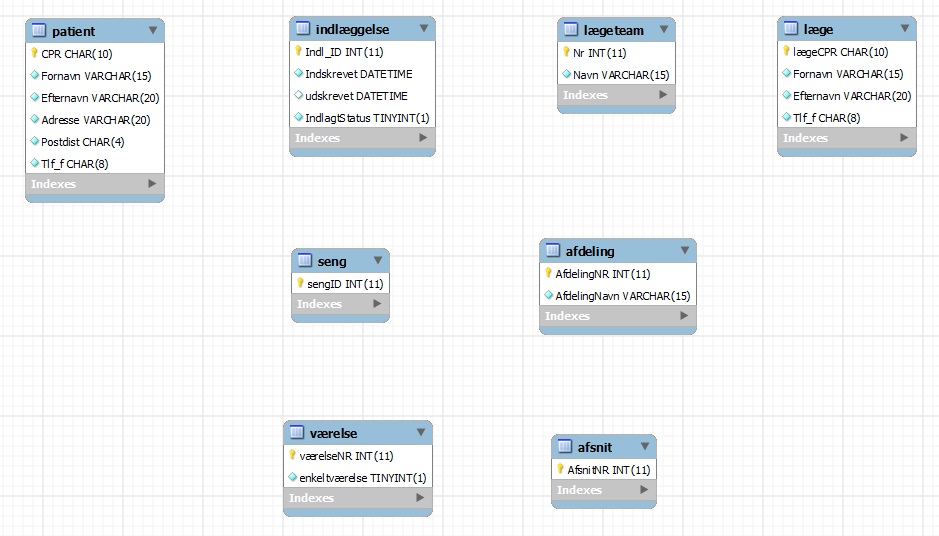
i entiteten værelse anvendes værelseNR som primærnøgle.

i entiteten afsnit anvendes afsnitNR som primærnøgle.

i entiteten afdeling anvendes afdelingNR som primærnøgle.

Script der opretter ovenstående entiteter kan findes i bilag 1.

## 2. Forbindelser ER-diagram



Tilknyttet

Tilknyttet

Tilknyttet

Tilknyttet

Tilknyttet

Tilknyttet

indlagt

Tilknyttet

Vi har udelukkende 1 : n forbindelser i vores ER diagram.

Vi tilføjer derfor primærnøglen fra n siden som fremmednøgle i 1 siden. Mellem patient og indlæggelse tilføjer vi primærnøglen indl\_ID som fremmednøgle i patient.

## 3. Konvertering

### 3.1 konverteringsalgoritmen

1. Svage entiteter gøres stærke.
2. For hver stærk entitet oprettes en tabel, med registrering af primærnøglen.
3. For hver 1:1 forbindelse tilføjes primærnøglen fra den ene tabel som fremmednøgle i den anden tabel.

Vi har ingen 1:1 forbindelser.

1. for hver 1:N forbindelse tilføjes primærnøglen fra 1-siden som fremmednøgle på N-siden.

Vi har i alt otte 1 : n forbindelser:

* Mellem patient og indlæggelse.
* Mellem indlæggelse og seng.
* Mellem seng og værelse.
* Mellem værelse og afsnit.
* Mellem afsnit og afdeling.
* Mellem afdeling og lægeteam.
* Mellem indlæggelse og lægeteam.
* Mellem lægeteam og læge.

Primærnøglen fra n siden, lægges over som fremmedsnøgle på 1 siden. For eksempelt tilføjes primærnøglen kunde\_id som fremmednøgle i entiteten reservation.

1. For hver N:M forbindelse oprettes en ny tabel med primærnøglerne fra de to tabeller som sammensat nøgle.

Vi har ingen n : m forbindelser.

1. For hver flerværdi attribut oprettes en ny tabel med primærnøglen og flerværdi attributten som sammensat nøgle. Samtidig fjernes flerværdi attributten fra den tabel den forekommer i.

Vi har to flerværdiattributter, i entiteterne patient og læge. For at kunne gemme flere telefonnumre på en læge eller en patient opretter vi derfor en ny tabel for hver entitet. telefonlæge og telefonpatient.

1. Hvis der findes forbindelser mellem mere end to entiteter, skal der oprettes en ny tabel med primærnøglerne fra alle de entiteter der indgår som sammensat primærnøgle.

Vi har ingen sådanne forbindelser.

### C:\Users\Anette Stidsing\Dropbox\1.SemesterOpgave\Database\EER diagram.jpg3.2 EER diagram

Script der viser konvertering findes i bilag 2

## 4. Data

Data indsættes i tabellerne.

Scriptet kan ses i bilag 3.

## 5. Test

use hospital;

### -- Test af foreign key

update patient

set cpr = '2566448125'

where cpr = '1205705322';

### -- delete, insert, update, select

delete from indlæggelse

where fk\_cpr = '1205705322';

insert into indlæggelse(indskrevet, indlagtstatus, fk\_CPR, fk\_sengID, fk\_lt\_nr)

values

('2012-08-08-14:30:00', true, '1205705322', 1, 1);

update patient

set fornavn = 'Niklas'

where fornavn = 'Lisbeth';

select \* from patient;

### -- Udtræk af data fra mere end en tabel

select patient.fornavn, patient.efternavn, indlæggelse.fk\_sengID, indlæggelse.fk\_lt\_nr, indlæggelse.indlagtstatus

from indlæggelse, patient

where indlæggelse.fk\_cpr = patient.cpr

group by indlæggelse.indskrevet;

# Programmering

## 1. Hvordan har vi valgt at løse opgaven

### 1.1 Cardlayot i modsætning til tabbed pane

Vi har valgt at bygge programmet op med cardlayout. Vi kunne dermed nemmere designe de forskellige sider som vi ønskede det, og samtidig give det en nem måde at bestemme hvilken side bruger skal være på. Et andet alternativ kunne være tabbed pane, hvor man har menuen med alle siderne og brugeren kan skifte frem og tilbage imellem de forskellige tabs. Det var bare ikke en løsning vi ønskede at bruge da vi ikke ønskede en synlig menu.

Vi har lagt vægt på at systemet skal være enkelt og brugervenligt. Brugergrænsefladen skal være simpel med et enkelt design således at brugeren ikke udsættes for flere valg end nødvendigt.

Vi har følgende cards i vores program.

* Hovedramme
  + Forside
  + Opret Patient
  + Patient info
    - Indlæg info
    - Indlæg patient
  + Dbinfo

### 1.2 GUI og brugervenlighed

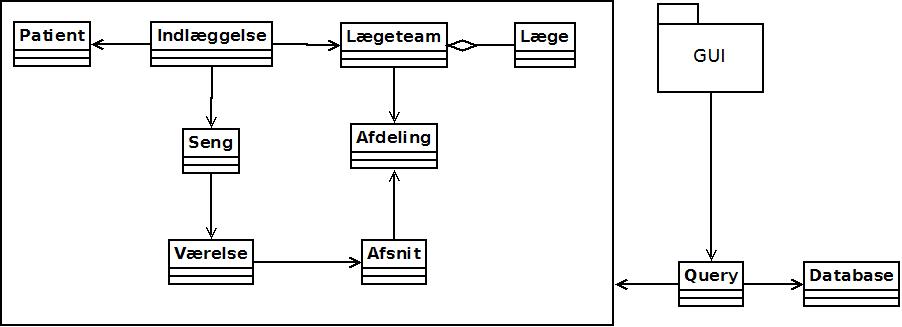
Vi har lagt vægt på at systemet skal være enkelt og brugervenligt. Brugergrænsefladen skal være simpel med et enkelt design således at brugeren ikke udsættes for flere valg end nødvendigt.

Vi har valgt at håndtere indlæggelse af en patient, således at afdeling, afsnit og seng er sat ind i comboboxes. Man skal vælge dem i rækkefølge, hvor man vælger afdeling først og seng til sidst.

I det virkelige liv, ville denne løsning give brugerne kunne få problemer med at overskue de forskellige afdelingsnumre og sengenumre og hvor de er på sygehuset.

En alternativ løsning kunne derfor være at lave en plan over sygehusets afdelinger og ledige senge i GUI'en, for at give brugeren et bedre overblik.

### 1.3 UML diagram



## 2. Klasser og centrale metoder

Vi har oprettet klasserne patient, indlæggelse, seng, værelse, afdeling, afsnit, lægeteam samt læge. Klasserne indeholderudelukkende get- og set metoder.

2.1 Database klassen

klassen Database, har ansvar for at håndtere forbindelsen til databasen. På den måde kan vi ved programmets start, oprette et database objekt og dermed oprette forbindelse til databasen. Det betyder, at hver gang vi skal hente eller indsætte noget i databasen, så kan vi bruge objektets metoder.

#### Connect()

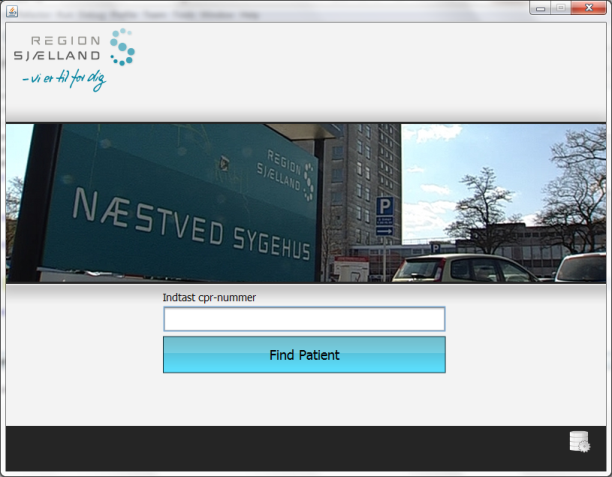
Metoden opretter forbindelse til de parametre som er medtaget fra constructoren (String db\_host, String db\_port, String db\_name, String db\_user, String db\_pw) og returnerer true eller false.

getData(String getdata)  
Bruger vi når vi skal hente data fra databasen. Så kalder man funktionen med en sql-streng så returnerer den resultatet i et resultset.

#### setData(String getdata)

Bruger vi når vi skal indsætte data i databasen. Så kalder man funktionen med den sql-streng der skal bruges til indsættelsen.

Database oplysninger  
Da ikke alle maskiner har samme database informationer, har vi valgt at lave et database card i selve programmet. Når programmet startes køres funktionen db.isConnected() som returnerer enten true eller false. Ud fra den returnerede værdi vises et af de to cards fremvist nedunder.



db.isConnected(false)

db.isConnected(true)

Hvis db.isConnected returnerer false er det først muligt at komme til søgning af patient, når der er oprettet forbindelse til databasen.

### 2.2 Query klassen

Denne klasse fungerer som en object handler. Den har ansvar for at hente data fra databasen og oprette objekter, som systemet kan arbejde med.

#### getIndlæggelse()

Metoden bruges til at hente info om indlæggelse af en person, og derefter returnerer et indlæggelsesobjekt. Vi udskriver derefter objektets data til vores indlæggelsesinfoside. Data hentes med sql'en:

Select indlæggelse.fk\_cpr, seng.sengID, indlæggelse.fk\_lt\_nr, indlæggelse.Indskrevet, indlæggelse.Udskrevet, indlæggelse.IndlagtStatus, afdeling.afdelingNr, afsnit.afsnitNr, værelse.værelseNr, lægeteam.navn

from Indlæggelse

inner join Seng on fk\_sengID = sengID

inner join Værelse on fk\_værelseNr = værelseNr

inner join Afsnit on fk\_afsnitNr = afsnitNr

inner join Afdeling on fk\_afdelingNr = afdelingNr

inner join Lægeteam on afdelingNr = fk\_afdelingsNr

where indlæggelse.fk\_CPR = CPR and indlæggelse.indlagtstatus = 1;

#### getPatient()

Bruges til at hente alle data fra patienttabellen hvor cpr nummeret passer med inputtet ned i et patientobjekt, og returnerer det. Data hentes med sql'en:

Select cpr, fornavn, efternavn, adresse, postdist from Patient where CPR = CPR;

#### getAfdeling()

Metoden bruges til at hente alle records fra afdelingstabellen og lægge dem ind i en arraylist af afdelingsobjekter. Data hentes med sql'en:

Select AfdelingNr, AfdelingNavn from afdeling;

### 2.3 Main klassen

#### jButton\_SøgCprActionPerformed(java.awt.event.ActionEvent evt)

Knappen tager input som er indtastet i søgefeltet, og forsøger at lave en query.getpatient() og lægge det ind i et patientobjekt. Derefter tjekker den om patientobjektet stadig er null. Hvis det eksisterer henter den patientens data ud, går over på infosiden og udskriver patientens info. Er patienten indlagt, vil den lave et query.getIndlæggelse() og lægge data ind i et indlæggelsesobjekt. Derefter vil den tage data fra indlæggelsesobjektet og lægge det over i indlæggelsesinfo.

#### jButton\_Opret\_patientActionPerformed(java.awt.event.ActionEvent evt)

Metoden forsøger at oprette en patient ved hjælp af query.createPatient() .

#### jButton\_indlægcard\_indlægActionPerformed(java.awt.event.ActionEvent evt)

Metoden kører afdelingsarraylisten igennem og sætter hvert enkelt objekt ind i comboboxen. Derefter sender den videre efter hentAfsnit() metoden.

#### jButton\_Patientstatus\_indlægActionPerformed(java.awt.event.ActionEvent evt)

Tager info fra comboboxene og patientobjektet og forsøger at oprette en indlæggelse ved hjælp af query.createIndlæggelse.

#### jButton\_Patientstatus\_udskrivActionPerformed(java.awt.event.ActionEvent evt)

Metoden bruges til at udskrive det valgte indlæggelsesobjekt og nulstiller alle info.

hentAfsnit(), hentVærelse() og hentSeng()

De enkelte metoder henter data fra den foregående combobox og sætter de objekter ind i den næste combobox som passer til.

## 3. Exception handling

#### 3.1 Tjekdata()

Når en bruger søger eller opretter en patient, skal patientens oplysninger indtastes i tekstfelter. Vi har derfor lavet en metode for at mindske risikoen for indtastningsfejl. Metoden tjekker om der er indtastet noget i tekstboksene. Desuden tjekker den om der er indtastet 4 tal i postdistrikt og om telefonnummer er på 8 tal. Hvis der ikke er korrekte oplysninger i tekstboksene, vil brugeren få en fejlmeddelelse og tjekdata er false. Hvis alle felter er korrekt udfyldt returneres true.

#### 3.2 Søge knap og opret knap

Når man indsætter noget i feltet til cpr nummer på forsiden og trykker på enten søg på patient eller opret patient knappen(ikke vist før man indtaster gyldigt cpr der ikke findes i databasen), tjekker programmet om teksten i tekstfeltet lever op til de krav vi har sat i databasen. Der kommer SQLExceptions hvis man enten lader feltet være tomt eller hvis alt inputtet ikke er tekst og vi har derfor en if sætning der tjekker for begge disse og udskriver den passende fejl besked i labelen over tekstfeltet. Udover dette har vi også et yderligere tjek af inputtet ved hjælp af en if sætning der tjekker om cpr nummeret er på 10 cifre, og udskriver derefter en fejlbesked på det førnævnte label hvis det ikke er på 10 cifre. Hvis inputtet er på 10 cifre der alle er tal vil den gå til info om patient hvis patienten allerede findes, eller komme med en besked der siger at nummeret ikke findes i databasen og vise opret knappen, som man så kan bruge til at oprette patienten med det cpr nummer der er skrevet i tekstfeltet.

#### 3.3 Oprettelse til databasen

Når vi åbner programmet vil den forsøge at oprette forbindelse til databasen ved hjælp af de forudbestemte data som vi har indsat i programmet, og hvis dette fejler vil programmet starte på database connection siden, hvor man kan indtaste user, password, host, port og database navn og derefter oprette forbindelse til databasen med de rigtige data. Hvis det ikke lykkes kan man ikke komme videre i programmet, da der alligevel ikke vil være nogle brugbare funktioner i programmet uden en aktiv forbindelse til databasen.

#### 3.4 Diverse fejl der kommer ved mistet forbindelse

Udover alle disse fejl vi har som kan løses ved hjælp af simple fejlmeddelelser til brugeren har vi også nogle fejl der kan ske hvis man mister forbindelse til databasen undervejs i programmet og stadig prøver at hente data ned fra databasen. Disse har vi valgt at udskrive på labels i bunden af programmet, som bare siger at det er en kritisk fejl og at brugeren skal kontakte en systemadministrator. Vi kunne også gøre så en systemadministrator så kunne trykke på en knap og så få en rigtig en fejlmeddelelse, evt. ved hjælp af en popup som så viser hele fejlmeddelelsen, som kun en it-kyndig kan bruge til noget, da vi ikke vil vise indviklede fejl til almindelige brugere af systemet og forvirre dem.

## 4. Test

### Oversigt Test Cases

|  |  |  |
| --- | --- | --- |
| Test id | Test navn | Status |
| 1 | Indlæg to patienter i samme seng | Bestået |
| 2 | Opret Patient | Bestået |
| 3 | Test at man kun kan indlægge en patient i en seng som lægeteamet er tilknyttet | Bestået |

### 4.1 Test 1 - Indlæg to patienter i samme seng

#### Beskrivelse:

Testen har til formål at teste om det er muligt i systemet at indlægge to patienter på samme seng.

#### Forudsætninger:

Adgang til databasen, samt have kørt de tre sql scripts

* 1\_Entiteter.sql
* 2\_konvertering.sql
* 3\_data.sql

#### Udførsel:

|  |  |
| --- | --- |
| Trin |  |
| 1 | Indtast følgende oplysninger i cpr nr. feltet:   * ” 2808852872”   Tryk find patient |
| 2 | Indlæg patienten ”Anette Stidsing” ved at klikke på knappen ”Indlæg”.  Det højre patient info panel skifter til indlæggelse med de mulige valgmuligheder for indlæggelse fremvist i combobokse. Knappen i venstre infoboks bliver sat til setVisible(false);  Vælg følgende oplysninger:   * Afdeling = ”Kirurgi” * Afsnit = ”7” * Værelse = ”20” * Seng = ”56”   Tryk Indlæg  Den højre patient info panel skifter tilbage til det forrige panel med de valgte indlæggelsesinformationer samtidig bliver en knap vist med teksten ”Udskriv” |
| 3 | Tryk på Tilbage knappen  Indtast følgende oplysninger i cpr feltet:   * ” 1301931901”   Tryk find patient |
| 4 | Indlæg Patienten ”Niklas Renner” ved at klikke på knappen ”Indlæg”  Vælg følgende oplysninger:   * Afdeling = ”Kirurgi” * Afsnit = ”7” * Værelse = ”20” * Seng = ”56”   Tryk Indlæg |

Forventet resultat:Det forventes, at det ikke er muligt at indlægge flere patienter i samme seng.   
Så når patient 1 indlægges oprettes der en record i databasen med indlæggelsesDato,fk\_CPR = ” 2808852872”, IndlagtStatus = ”1”, fk\_SengID=”56” samt fk\_lt\_nr = ”3”.

Når patient 2 skal indskrives kan der ikke vælges den samme seng fordi den allerede er optaget i databasen.

#### Faktiske resultat

Brugeren får ikke mulighed for at vælge seng 56, men kun de andre ledige senge på værelset. Hvilket betyder at sengen allerede er optaget.

#### Konklusion på test casen

På baggrund af testen kan man konkludere, at det faktiske resultat stemmer overens med det forventede resultat. Der kommer ingen fejl beskeder, når brugeren prøver at dobbelt book, fordi vi fjerner muligheden for det ved at fjerne sengen fra comboboksen efter første indlæggelse. Så programmet virker efter hensigten

|  |
| --- |
| **Bestået** |

### 4.2 Test 2 - Opret patient

Opret en patient i systemet.

#### Beskrivelse:

Testen har til formål at teste om det er muligt at oprette en patient i systemet.

#### Forudsætninger:

1. Der må være adgang til databasen og følgende scripts skal køres:

1\_Entiteter.sql

2\_konvertering.sql

3\_data.sql

2. CPR nummer skal være på 10 tal og må ikke indeholde bogstaver.

3. Fornavn skal indeholde mindre end 15 tegn.

4. Efternavn skal indeholde mindre end 20 tegn.

5. Adresse skal indeholde mindre end 20 tegn.

6. Postdist skal indeholde 4 tal.

7. Telefonnummer skal indeholde 8 tal.

#### Udførsel:

|  |  |
| --- | --- |
| Trin |  |
| 1 | Indtast følgende oplysninger i cpr nr. feltet:   * ” 2606852626” |
| 2 | Tryk på knappen "find patient" |
| 3 | Følgende meddelelse vil derefter vises: "CPR-nummer findes ikke i databasen!"  En ny knap "opret ny patient" vil synliggøres. |
| 5 | Tryk på knappen "opret ny patient". |
| 6 | Udfyld teksfelterne på opret patient-siden således:   * Fornavn: Mia * Efternavn: Larsen * Adresse: Søndergade 17 * Postdist: 8981 * Tlf: 54536695 |
| 7 | Tryk på knappen "opret patient" |
| 8 | Patient info vises derefter i et nyt skærmbillede, hvor alle patientens oplysninger er skrevet ud. |

Forventet resultat:Det forventes at der fremkommer et skærmbillede "patient info". Patientens info (CPR, fornavn, efternavn, adresse, postdist samt tlf) er udskrevet i felterne på skærmbilledet.

Det forventes at der oprettes en record i tabellen patient i databasen, med følgende indhold:

CPR =” 2606852626”,fornavn="Mia",efternavn="Larsen",Adresse="Søndergade 17",postdist="8981",Tlf="54536695"

#### Faktiske resultat

Der fremkommer et skærmbillede "patient info". Patientens info (CPR, fornavn, efternavn, adresse, postdist samt tlf) er udskrevet i felterne på skærmbilledet.

Der oprettes en record i tabellen patient i databasen, med følgende indhold:

CPR =” 2606852626”,fornavn="Mia",efternavn="Larsen",Adresse="Søndergade 17",postdist="8981",Tlf="54536695"

#### Konklusion på test casen

Testen gav det forventede resultat.

|  |
| --- |
| **Bestået** |

### 4.3 Test at man kun kan indlægge en patient i en seng som lægeteamet er tilknyttet

TC3 - Indlæg en patient i en seng som lægeteamet ikke tilknyttet.

**Beskrivelse:**  
Testen har til formål at teste om det er muligt i systemet at indlægge en patient i en seng fra andre lægeteams end den oprindeligt er knyttet til.

**Forudsætninger:**

Adgang til databasen, samt have kørt de tre sql scripts

* 1\_Entiteter.sql
* 2\_konvertering.sql
* 3\_data.sql

Udførsel:

|  |  |
| --- | --- |
| Trin |  |
| 1 | Indtast følgende oplysninger i cpr feltet:  ”2808852872”  Tryk find patient |
| 2 | Du vil se vinduet ”Patient Info”  Indlæg Patienten ”Anette Stidsing” ved at klikke på knappen ”Indlæg”  Tryk ”Indlæg” |
| 3 | Du vil nu få mulighed for at tildele indlæggelses oplysninger (afdeling, afsnit, værelse og seng)  Vælg følgende oplysninger: Afdeling = ”Hjerte”  Afsnit = ”2”  Værelse = ”4”  Seng = ”10” |
| 4 | Nu hvor oplysningerne er indtastet, vil vi se at det ikke vil være muligt at vælge en seng fra et andet Læge Team, når der skiftes Afdeling:  Vælg følgende:  Afdeling = ”Lunge”  De følgende oplysninger vil på forhånd skifte til:  Afsnit = ”4”  Værelse = ”10”  Seng = ”28” |

**Forventet resultat:**Det forventes at det ikke er muligt at tildele en seng fra et andet læge team som sengene er knyttet til.  
Dvs. når der tildeles en afdeling for patienten, så vil der være specifikke afsnit, værelser og senge, som er knyttet til lige netop den afdeling samt læge team.

Hver læge team er knyttet til sin engen afdeling på forhånd, så der kan ikke vælges et læge team og en anden afdeling, så man risiker ikke at tildele en seng til en anden afdeling/læge team.

Det forventes at hvis man ændre på feltet ”Afdeling” så skifter ”Læge Team” automatisk afhængigt af valget. Dvs. ”Afdeling – Hjerte” er knyttet til ”Læge Team – Hjerte Team” og ”Afdeling - Lunge” er knyttet til ”Læge Team – Lunge Team” og ligeledes med Kirurgi. Disse afdelinger/teams er tildelt hver i ser bestemte afsnit, værelser og senge hierarkisk. Derfor vil det ikke være muligt at vælge en seng som Læge Teamet ikke står til ansvar for.

**Faktiske resultat**

Vi ser at hver gang vi ændre på feltet ”Afdeling” så skifter ”Læge Team” automatisk afhængigt af valget. ”Afdeling – Hjerte” er knyttet til ”Læge Team – Hjerte Team” og ”Afdeling - Lunge” er knyttet til ”Læge Team – Lunge Team” og ligeledes med Kirurgi. Det ikke er muligt at vælge en seng som Læge Teamet ikke er knyttet til.

**Konklusion på test casen**

På baggrund af den samlede test konkluderes, at programmets status er funktionel. Der opstår ingen fejl og dermed er testen bestået.

|  |
| --- |
| **Bestået** |

# IT i organisationen

### 1. Organisationsmodel, der beskriver ansvar myndighed, arbejdsdeling og redegør for de funktioner der skal være.

For at kunne behandle patienterne bedst muligt, må der for hver afdeling være en klar fordeling af myndighed og ansvar. Organisationsmodellen (bilag 5) følger derfor linjeprincippet.

Hospitalsdirektøren har det overordnede ansvar for virksomhedens drift, behandling og pleje. Lige under ham er der tre øvrige chefer: en sygeplejerske faglig chef, en lægefaglig chef og en administrativ chef. Den administrative chef har ansvar for virksomhedens service- og støtte funktioner, som består af en økonomi afdeling, en IT afdeling og en teknisk afdeling.

De teknisk- og administrative medarbejderes opgave er at levere støtte til læger, patienter, besøgende og generelt ansatte på sygehuset. Den administrative service omfatter indsigt i juridiske og økonomiske emner, samt koordinering af møder og præsentation af resultater. IT afdelingen yder support og sørger for at sygehusets systemer og it del kører.

Under hver afdeling er der en overlæge, en oversygeplejerske og en række lægeteams bestående af fire læger. Overlægernes og oversygeplejerskernes opgave er at sørge for koordinering og arbejdsdelingen i afdelingen. De må ligeledes være velinformeret og i kontakt med de andre afdelinger, for bedst muligt at kunne koordinere arbejdsdelingen. Da Næstved sygehus er en større mere kompleks organisation med mange ansatte, er det derfor en fordel at have afdelingerne som fungerer som små selvstændige enheder, med en overordnet som har ansvar for koordinering og arbejdsdeling for hver afdeling.

Patienterne vil blive tilknyttet et fast lægeteam som vil være ansvarlig for patientens behandling under indlæggelsen. I hvert lægeteam er der udvalgt en teamleder, som har ansvaret for at forløbet for patienternes indlæggelse går glidende. Dette kunne f.eks. være i forbindelse med, at patienten skal flyttes til en anden afdeling.

Hver medarbejder har altså en direkte overordnet, således at man undgår modstridende ordrer, og undgår forvirring i forbindelse med behandling og medicinering af sygehusets patienter. Kommunikationen vil derfor være overskuelig, hvilket er nødvendigt på et sygehus.

Arbejdsdelingen på sygehuset følger funktionsprincippet. Hver afdeling har medarbejdere som er specialiserede indenfor et bestemt område.

### 2. Hvordan sikrer ledelsen sig at IT-sikkerheden overholdes i forbindelse med patientjournalen?

For at ledelsen kan sikre sig at IT-sikkerheden bliver overholdet, så er det en nødvendighed at forstå persondataloven, IT-Politik og kommunikationssikkerhed og evt. andre relevante love rettigheder og standarder, for at systemet kan effektiviseres mens loven/politikken bliver overholdt.

Der er mange former for IT-sikkerhed, som er en stor kæde af forhindringer imod nogle eller flere oplysninger bliver krænket, og der har IT-politikken en standard for IT-systemer. Et hvert IT-system skal sikre ***Fortrolighed***, ***Tilgængelighed*** og ***Pålidelighed***, derudover er der sikkerhedsproblemer som er opdelt i fire hovedtemaer, ***Fysisk*** ***sikkerhed***, ***Logisk*** ***sikkerhed***, ***Kommunikationssikkerhed*** og ***Datasikkerhed***.  
Ved fysisk sikkerhed forstår vi som en hindring mod tyveri eller fysisk skade, og derfor mener vi at det er relevant at tildele læge og sygeplejerske personalet med et kort/chip for aflåsning/oplåsning af disse lokaler, det kan evt. være lægekontor eller datarummet.  
Ved logisk sikkerhed forstår vi som begrænsning af brugeradgang, med evt. log ind navn og kode.  
Det er vigtigt at tildele læge og sygeplejerske personalet ét hver, med de relevante brugeradgange. Fordi sygeplejersken ikke skal have rettigheder til at ændre noget på patientjournalen, som lægen har udfyldt, ved dette skaber man et hierarkisk system og orden, der formindsker evt. fejlmedicinering, kommunikationsmangel, og der tages forbehold for pedeler o.l.  
Ved kommunikationssikkerhed forstår vi ved at sikre WAN/LAN, at kommunikationslinjer ikke giver uautoriseret adgang til data og/eller dokumentforfalskning. Der er flere muligheder for at sikre sig mod uautoriseret adgang, og det kan bl.a. være Kryptering, Firewall, lukkede linjer. Herunder vil it-afdelingen jævnligt opdatere og vedligeholde systemer og hardware til de nyeste opdateringer, eftersom trusler mod spyware, vira og hackere ikke ophører. Hvis dette ikke overholdes kan det have katastrofale konsekvenser for et sygehus, som hardwareoverophedning og direkte lukning af systemet, eller elektronisk tyveri af journaler. Udover at installere de basale anti-virus/spyware og firewall programmer på pc’erne så er det også en god ide med en VPN linje til kollegaernes pc’er. En VPN står for Virtual Private Network, som opretter en tunnel mellem medarbejdernes pc. Sikkerheden ved VPN består af autentificering, kryptering og integritet, disse features kaldes også SSL protokol - Secure Socket Layer - som benyttes af alle internet baserede programmer, så kollegaerne kan stole på de oplysninger der bliver sendt via linjen, altså at oplysningerne har en troværdighed, og at det er en autoriseret der har tilsendt og/eller modtaget oplysningerne.  
Ved datasikkerhed forstår vi som opbevaring og intakt af data, så det ikke forsvinder eller forvanskes. Her er der tale om backupprocedure, hvor der tages hensyn til hyppigheden, omfanget og antal af kopiversioner. Dette er essentielt for hospitalet, for at sikre sig sikkerheden af patientjournaler, hvis der nu i et tilfælde af nogle patientjournaler forvanskes eller beskadiges. Hvis man så har været uheldig nok til tage en backup af en beskadiget journal, så kan man have glæde af, at man har taget backupper jævnligt, eller hvis selve backuppen er beskadiget, så kan man benytte de andre versioner af samme backup.   
Ved katastrofeplan forstår vi af opbygning af en procedure der er med til at hjælpe og holde overblikket så man er klar på at stoppe ulykken og forbyg den hurtigst muligt. Dette uddrages fra en risikoanalyse, hvori man inddrager de mest effektive, nødvendige og hurtigste løsninger. Dog bruges den også til at finde svagheder på virksomheden, og i vores tilfælde er det et hospital.

Ved at gøre nytte at disse programmer, funktioner og hardware kan ledelsen sikre sig at IT-sikkerheden, samt persondataloven overholdes, ved at kun give adgang til autoriseret personale på hospitalet, af behandling følsomme data. I det her tilfælde kan der være tale om CPR, navn, adresse, tlf., sygedomme og hvad der ellers står stå på en patientjournal.

### 

### 3. I forbindelse med anskaffelsen af det nye system bedes I redegøre for hvilken måde ledelsen bør implementerer systemet for de ansatte, så alle bakker op om det nye system.

Der bør først udarbejdes en systemarkitektur for at fastlægge en åben og fleksibel informationsteknologisk infrastruktur så teknologiens strategi kan udnyttes bedst muligt, så computere, database og netværk fungerer fornuftigt sammen. Ansatte vil aldrig godtage eller gøre brug af systemer der er for komplekse eller besværlige. Programmer og hardware skal være til for at simplificere og effektivisere arbejdspladsen, derved skaber man både et større overskud og et bedre arbejdsmiljø for de ansatte. Dvs. hvis vi implementer vores hospitalssystem, vil det kun være lettere for læger mm. at indskrive og udskrive samt søge på patienter, ved tryk af fingerspidsen. Derfor mener vi, at vores system vil overholder de anssattes behov for simplificere arbejdet ved at benytte programmet, og vil bakke om det nye system. Det kan evt. tages i brug i trinvis, så det er kun en afdeling der benytter systemet, med mindre kritisk skader/ikke livstruende, i stil af røntgenafdeling, og så kan man ligeså småt sprede systemet videre til de andre afdelinger. På den måde kan man tage stilling til opdateringer eller justeringer der skal foretages til den kommende opdatering, og på samme tid kan man få den nødvendige feedback af brugeren, så systemet kan skræddersys for hospitalet.

### 4. Redegør hvilke fordele der kan være for patienterne og de ansatte ved indførelsen af patientjournalen

Som beskrevet i opgaven, har der på Næstved sygehus været flere tilfælde af fejlmedicinering. Dette er sket på grund af mangel på kommunikation mellem medarbejderne. Patientjournalen vil kunne anvendes både til information, kommunikation og oprettelse af planlagte handlinger. Læger og sygeplejerske vil altså kunne hente oplysninger om patientens indlæggelser, eventuelle sygdomme, hvilken medicin patienten tager og tilføje hændelser og information om patientens behandling.

I stedet for tunge patientjournaler vil læger og sygeplejerske ved indførelse af den nye patientjournal hurtigt kunne få et overblik over den enkelte patients historik. Dette vil optimere både planlægning og patientsikkerheden, og der vil derfor være mindre risiko for fejlmedicinering. Det vil lette de ansattes arbejde og spare dem kostbar tid, Da de ikke skal bladre i en stak papirer i flere minutter for at finde de nødvendige oplysninger på en patient. Læger og sygeplejerske vil opleve en mindre stresset hverdag og samtidig føle sig mere trygge i deres arbejde.

Da patienterne er afhængige af det offentlige ved indlæggelse, vil også de kunne føle sig mere trygge under indlæggelse. De vil opleve en mere sammenhængende behandling, hvor de læger og sygeplejerske som er tilknyttet patienten, er oplyste om alle hændelser og information omkring patienten.

### 5. Vurder konsekvenserne ved at bruge det nye system, dels økonomisk men også organisatorisk.

Det nye system vil lette en administrative arbejdsbyrde. Det nye system vil spare personalet tid, da de hurtigt vil kunne slå patienten op. Da lægerne vil kunne se patientjournalen mens de taler med de enkelte indlagte patienter, behøves de dermed ikke huske patientens info og der vil derfor være mere tid til at behandle patienterne.

# Ansvarsområder

## Systemudvikling

|  |  |
| --- | --- |
| Områdenavn | Ansvarlig |
| Brugsmønstre | Anette |
| Tilstandsdiagram | Niklas |
| Hændelsestabel | Martin |
| Aktørtabel | Thomas |
| Funktionsliste | Anette |
| Brugergrænseflade | Martin |

## Database

|  |  |
| --- | --- |
| Områdenavn | Ansvarlig |
| Oprettelse af entiteter | Martin |
| ER diagram | Anette |
| Data | Thomas |
| Test | Niklas |

## Programmering

|  |  |
| --- | --- |
| Områdenavn | Ansvarlig |
| GUI design | Thomas |
| Test case 1 | Thomas |
| Test case 2 | Anette |
| Test case 3 | Martin |
| Centrale metoder i Query klassen | Niklas |

## IT i organisation

|  |  |
| --- | --- |
| Områdenavn | Ansvarlig |
| Organisationsmodel | Anette |
| Sikkerhed i forhold til patientjournal | Martin |
| Fordele ved patientjournal | Anette |
| Økonomiske og organistoriske konsekvenser | Martin |