Testprojekt

Test af ordinationssystem

Erhvervsakademi Aarhus
Professionsbachelor i It-arkitektur
April 2025

Fællesprojekt imellem Systemudvikling og Softwarearkitektur på 3. semester

Testprojekt – 3. semester
Professionsbachelor i It-arkitektur

Projektbeskrivelse

Projektet går ud på at færdigudvikle og teste et system til medicinordinering.

Målene for projektet er

- At færdigimplementere en model, der kan bruges til at registrere forskellige typer af ordinationer af lægemidler til patienter.
- At lave unit test af den implementerede model.
- At lave en testspecifikation for use-casen ”Opret ordination” samt gennemføre en test ud fra testspecifikationen.

Beskrivelse af case

Et sygehus har behov for at vide hvilken og hvor meget medicin der gives, til de forskellige patienter. En ordination kan være en af følgende 3 typer.

- Daglig fast
- Daglig skæv
- Efter behov (PN: pro necesare)

For alle ordinationer gælder, at de har en start- og en slutdato, og at ordinationen er gældende i den pågældende periode.

Ved **daglig fast** forstås, at medicinen gives dagligt, i den periode ordinationen er gyldig. Med **fast** menes, at medicinen højst gives 4 gange i døgnet på forud bestemte tidspunkter (morgen, middag, aften og nat). For hvert af disse tidspunkter registreres, hvor mange enheder patienten skal have af den pågældende medicin.

Med nedenstående eksempel på en registrering, skal patienten have medicinen morgen, middag og nat, og der gives hhv. 2, 1 og 1 enhed af medicinen.

Morgen	Middag	Aften	Nat
2	1		1

Ved **daglig skæv** forstås, at medicinen gives dagligt, i den periode ordinationen er gyldig. Med **skæv** menes, at man ved ordinations oprettelse bestemmer, hvilke klokkeslæt medicinen skal gives på. Når ordinationen oprettes, angives det, hvor meget og på hvilke tidspunkter af dagen patienten skal have medicinen. Her kan angives et vilkårligt antal tidspunkter på døgnet, og for hvert tidspunkt, angives antallet af enheder, patienten skal have af den pågældende medicin.

Med nedenstående registrering, skal patienten have medicinen seks gange i døgnet på de angivne klokkeslæt:

09:30	10:30	13:30	14:30	19:30	20:30
2	1	2	1	2	1

Ved **PN** forstås, at medicinen gives efter behov, i den periode ordinationen er gyldig. Her registrerer man, hvor mange enheder patienten skal have af medicinen, når der er behov. På en PN-ordination holdes der rede på, hvilke dage patienten har fået medicinen. Bemærk, at patienten godt kan få medicinen flere gange på samme dag, dvs. den samme dato kan registreres flere gange.

De enheder medicinen kan gives i er: Styk, pust, ml eller dråber.

For en vilkårlig af de tre ordinationstyper, skal det være muligt at beregne såvel døgndosis som den samlede dosis af den pågældende medicin. Der skal altid regnes i hele dage, hvor start- og slutdato er inklusive.

For PN skal døgndosis beregnes som:

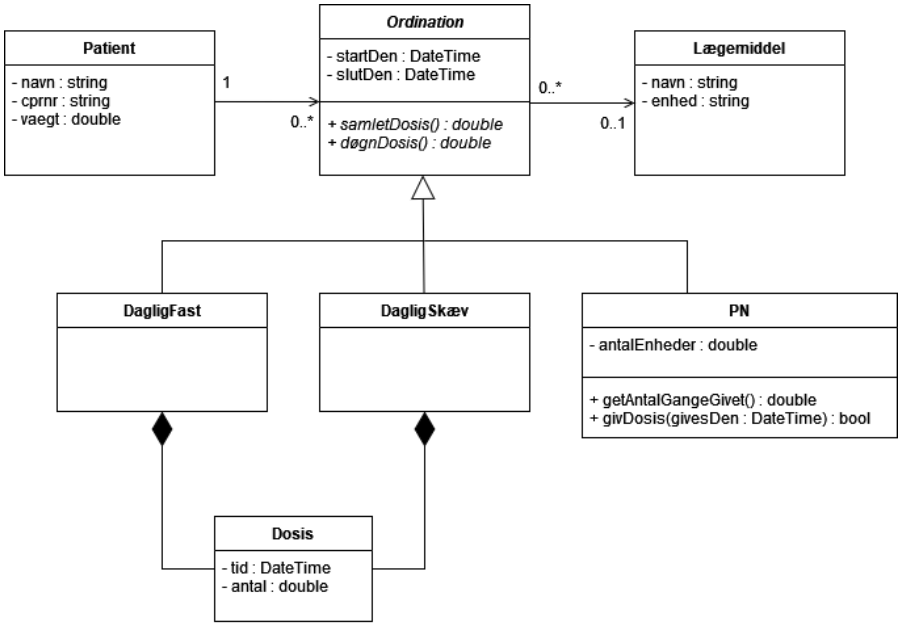
*(antal gange ordinationen er anvendt * antal enheder) / (antal dage mellem første og sidste givning)*

Når man har en patients vægt og et lægemiddel, skal systemet kunne beregne de anbefalede antal enheder pr. døgn af det pågældende lægemiddel til patienten. De anbefalede antal enheder er afhængig af patientens vægt. Der bruges tre forskellige faktorer til beregningen, afhængig af om patienten vejer mindre end 25 kg (let), mere end 120 kg (tung) eller derimellem (normal). Vægt og faktor ganges sammen.

Ekstraopgave – udfordring: Til opfølgning på forbruget af lægemidler, skal systemet kunne beregne antal af ordinationer, der er givet med et bestemt lægemiddel, til patienter der har en vægt inden for et givet interval.

Udviklingsarbejde

Her er et ufuldstændigt designklassediagram for modelklasserne i systemet. Der henvises til implementationen for de specifikke detaljer.



Mange af klasser er allerede realiseret, men nogle mangler at blive lavet helt færdige. Dette gælder konkretiseringerne af Ordinations-klassen, samt dele af den abstrakte Ordinations-klasse og ikke mindst Service-klassen, som implementerer funktionalitet til API'en.

Blazor-delen leveres færdigkodet, men vil muligvis kunne give fejl visse steder indtil API'en er helt på plads.

Selve projektet kan findes på Canvas som .zip. Bemærk, at delprojekterne heri er målrettet til .Net 6.0, som I bedes anvende.

Læs koden og se hvad der mangler i modellen og Service-klassen. Herefter skal I færdigudvikle de manglende metoder.

Der skal udvikles "unit test" af udvalgte metoder fra klasserne: DataService (API), Ordination, DagligFast, DagligSkæv og PN. Testene skal herefter programmeres med MStest i projektet "ordination-test".

Faglige mål for systemudvikling

- At kunne finde ækvivalensklasser med baggrund i metoders signatur med blackbox-teknikken
- At kunne bestemme konkrete testdata med baggrund i fundne ækvivalensklasser samt grænseværdier (når det giver mening)
- At kunne specificere en test med baggrund i en use case-specifikation

Faglige mål for softwarearkitektur

- At kunne færdigimplementere modelklasser ud fra en specifikation og et diagram
- At kunne færdigimplementere forretningslogik i en Service-klasse
- At kunne anvende MStest til at lave unit test i et .NET projekt
- At få mere programmeringserfaring med C# og .NET

Processen

I skal arbejde i grupper af højst 3 personer

Tidsplan:

1. Programmér systemet (**23. April, softwarearkitektur**)
2. Lav og dokumentér testcases til unit test af klassemodellen ved anvendelse af blackbox-test teknikken (**28. April, systemudvikling**)
3. Lav testspecifikation til use casen "Opret ordination" (**28. April, systemudvikling**).
4. Realisér, dvs. kod unittestene ud fra testcasene under anvendelse af MStest (**5. Maj, softwarearkitektur)**
5. Hver gruppe anvender en af de andre gruppers testspecifikation til use casen "Opret ordination" for at teste den anden gruppes system (**5. Maj, softwarearkitektur**).

Aflevering

Projektet er en obligatorisk aflevering. Aflevering skal foregå senest **6. Maj klokken 23.59** på Canvas.

I skal i softwarearkitektur aflevere et link til GitHub med alt jeres kode.

I skal i systemudvikling på Canvas aflevere et testdokument med eksempler på testcases til blackbox test af centrale klassers metoder samt testcases til test af use casen, samt resultatet af use case testen udført af en anden gruppe.

Fremlæggelse og evaluering

I skal klargøre en fremlæggelse på ca. 7 minutter til den **12. Maj (softwarearkitektur)**, der

- for en kompleks metode viser hele testforløbet fra planlægning til udførelse
- viser elementer af jeres testspecifikation for use casen ”Opret ordination”
- viser jeres realisering af klasserne DagligFast, DagligSkaev og PN
- en kort rapportering af forløbet, hvor en anden gruppe testede jeres program med jeres specifikation

Ved spørgsmål fra medstuderende eller lærere skal I hurtigt kunne finde og vise andre dele af testdokumentation eller kildetekst.

Use Case Name:	**Opret Ordination**	
Scenario:		
Trigger Event:	Patient er diagnosticeret og skal have en medicinsk behandling	
Brief Description:		
Actors:	Sygeplejerske	
Related use cases:		
Stakeholders:	Patient, læge, sygeplejerske	
Precondition:	Patient og lægemidler er registreret i systemet	
Postcondition:		
Flow of events:	**Actor** 1. Find patient frem ud fra cpr-nr 2. Vælg lægemiddel 3. Vælg typen af ordination og gå til opret 4. Hvis ej PN beslut ud fra beregnet dosisPrDøgn hvor mange dosis og med hvilke antal enheder på doserne, der skal oprettes. 5. Hvis dagligFast angiv start og slutdato og de 4 gange antal enheder og vælg opret doser. Hvis PN angiv start og slutdato, angiv antal enheder. Hvis dagligSkæv angiv start og slutdato og så længe der ønskes oprettet flere og den anbefalede dosisPrDøgn ikke overstiges, angiv antal enheder og en tid og vælg opret dosis. 6. Afslut oprettelsen	**System** 3.1 Beregner anbefalet dosisPrDøgn og viser det på skærmen 5.1 Hvis dagligSkæv holder systemet øje med den samlede mængde af de indtil nu oprettede doser og viser dem på skærmen. Der gives besked hvis anbefalet dosis overskrides.
Exceptional Flows:	5 Hvis startdato større end slutdato så gives fejlmeddelelse	