Forprosjektrapport

System for behandling av XML-meldinger

av Ola G. Berg, Tom H. Basmo, Nikola Dordevic, Jørund T. Løvlien & Hajin Barzingi

OSLOMET

Dato: 24.01.2022

Versjon: 1.2

Sammendrag

Gruppen består av fem dataingeniør-studenter som har fått et oppdrag fra Kreftregisteret, med Jan Franz Nygård, leder for Registerinformatikkavdelingen, som kontaktperson.

Oppdraget går ut på å lage et nytt system for å ta imot, vise og redigere XML-meldinger som blir sendt inn fra Norsk Helsenett. Kreftregisteret har allerede et slikt system, men teknologien som er brukt er utdatert og vanskelig å integrere med andre systemer. Registerinformatikkavdelingen har gjort undersøkelser tidligere og kommet frem til at en ny løsning med teknologiene Java EE, React, SurveyJS og Hibernate vil være den beste. Vi har undersøkt teknologiene som er anbefalt og også sett på alternativer. Basert på analysene, Kreftregisterets ønske og kompetanse innenfor de ulike teknologiene, er React og SurveyJS valgt som teknologi for å kode frontend og brukergrensesnitt, mens Java med Spring Boot og Hibernate er valgt for å kode backend-løsningen. Java Spring Boot er et rammeverk basert på Spring som gjør det enkelt å utvikle REST API og mikrotjenester. Valget falt på denne teknologien fordi det skal være greit å skalere og på grunn av “AutoConfigure” er raskt å komme i gang med.

Løsningen som blir utviklet for Kreftregisteret vil blant annet bidra i arbeidet med å modernisere et utdatert system og være enklere å vedlikeholde og videreutvikle. I tillegg vil løsningen effektivisere arbeidet til de medisinske koderne.

Innhold

1. Presentasjon

1.1. Gruppen

Gruppen består av følgende fem dataingeniør-studenter:

- Ola G. Berg
  s341874
- Tom H. Basmo
  s340432
- Nikola Dordevic
  s341839
- Jørund T. Løvlien
  s341822
- Hajin Barzingi
  s238727

1.2. Oppdragsgiver

Oppdragsgiver er Kreftregisteret. Kreftregisteret ble opprettet i 1951 og er et av de eldste nasjonale kreftregistre i verden. I tillegg til forskningsarbeid og ansvar for screeningprogrammer, samler de inn data og lager statistikk over alle krefttilfeller i Norge.

Vår kontaktperson ved Kreftregisteret er leder for registerinformatikkavdelingen, Jan Franz Nygård.

1.3. Beskrivelse av prosjekt

Kreftregisteret ønsker et web/API-basert system som kommuniserer med resten av fagsystemene deres, og som viser det medisinske innholdet fra XML-meldinger i en webleser, der innholdet kan redigeres og sendes tilbake til fagsystemet.

For å løse oppgaven, har vi visualisert den i fem underoppgaver:

1. Ta imot en XML-fil => Java
2. Lese den opp i et web GUI => React
3. Ha mulighet til å endre => Surveyjs
4. Validere => mot XSD og mot et API-kall mot metadatabasen ved Kreftregisteret
5. Lagre som en XML => Hibernate

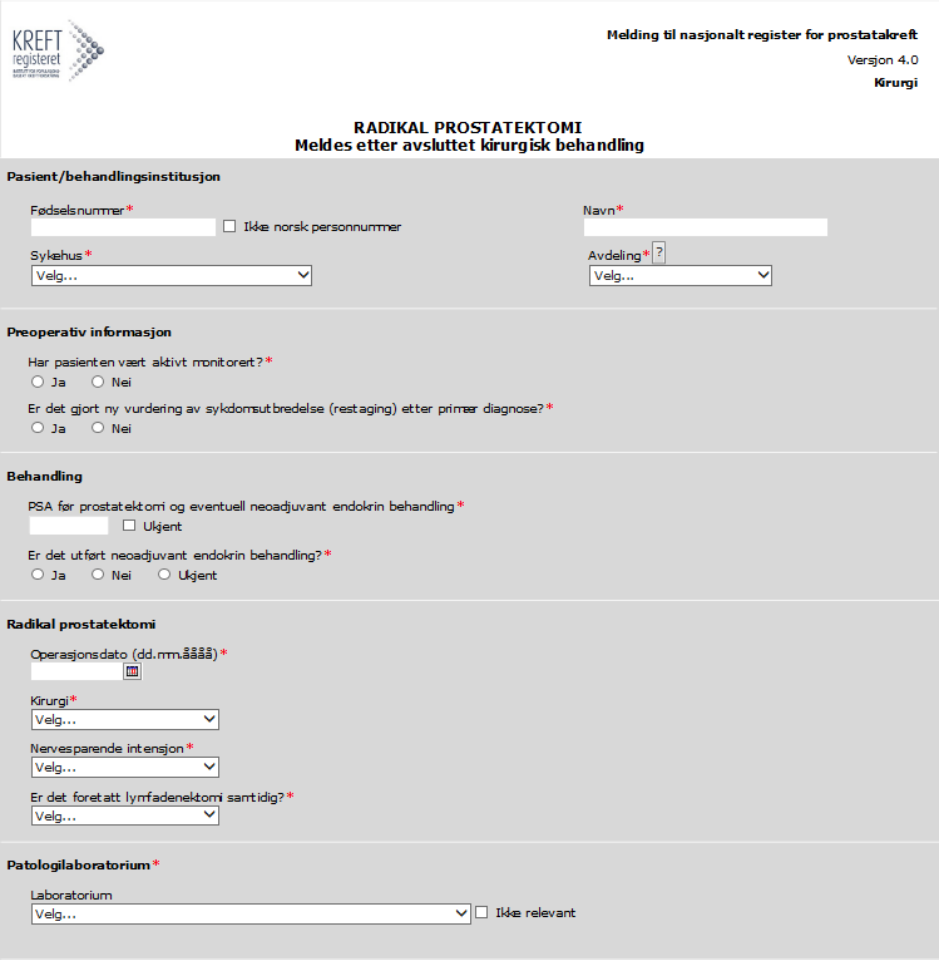
Målet vårt er å lage en løsning som fungerer for én enkel krefttype (muligens flere). Verktøyene vi har sett oss ut er React og Surveyjs i front-end, og Java Spring Boot med Hibernate i back-end.

1.4. Framdriftsplan

Developement schedule - Rough																						
Week	Main focus	1	2	3	4	5	6	7	9	10	11	12	13	14	16	17	18	19	20	21	22	23
1-3	Research XML, SurveyJS, given documents from client. Write "forprosjektsrapport". Workshop with client. Set up a YouTrack/Trello for Scrum-method. Define MVP for project.																					
4-7	Outline documentation (which will be written as we go along). Create use cases based on the MVPs and appoint prioritization. Appoint responsibilities. Set up detailed schedule.																					
8	Winter break																					
9-14	Improve and add to product through iterative process. Usertest product.																					
15	Easter break																					
16-20	Improve and add to product through iterative process. Final touch-up documentation and finish project report.																					
21	Hand in final product																					
22	Prepare presentation																					
23	Presentation of product																					

2. Dagens situasjon

I helsevesenet sendes pasientinformasjon mellom sykehus og fra sykehus til Kreftregisteret i form av XML meldinger i et eget helsenettverk (Norsk Helsenett). Kreftregisteret mottar flere hundre tusen slike meldinger hvert år.



Figur 1: Skjermbilde av dagens løsning med InfoPath

I dag brukes det et kommersielt program (InfoPath fra Microsoft) til å lese og vise XML meldingene til de medisinske koderne. InfoPath er "end of life" og videreutvikles ikke lenger. Det er også et nokså "frittstående" program, og kommuniserer ikke godt med Kreftregisterets andre fagsystemer som er Java-baserte.

3. Mål og rammebetingelser

Målet er å lage et system som:

- Tar imot XML-meldinger
- Viser dataene fra meldingene i et grensesnitt
- Muliggjør redigering av data fra grensesnittet
- Validerer endret data
- Kommuniserer med resten av fagsystemene til Kreftregisteret som er Java-baserte
- Er smidig og effektiv å bruke for brukerne

Kreftregisteret har gitt noen rammebetingelser, slik som at løsningen kun skal bestå av Open-Source teknologier, nettleserbasert front og at systemet skal være skalerbart for å kunne bli brukt med flere krefttyper. I første omgang skal løsningen kun fokusere på en krefttype, prostata. Løsninger for hvordan systemet skal få inn XML-meldingene eller hvor de skal lagres (database), skal ikke utvikles da dette i dag blir gjort av andre fagsystemer.

Utviklingen vil være modulbasert, i det tilfellet "Proof of Concept" er vellykket og Kreftregisteret har lyst til å ta i bruk og videreutvikle systemet.

Kreftregisteret har tidligere gjort egne undersøkelser og sett på løsninger for et nytt system, og løsningsmodellen som foreslås å bruke er React og SurveyJS til front-end, Java EE og Hibernate til back-end. React er en teknologi de allerede bruker på andre områder og systemene er stort sett utviklet med Java. De er åpne om å bruke andre teknologier enn SurveyJS og Java EE, så lenge de er Open-Source. I kapittel 4 kan en lese om de andre teknologiene som ble vurdert og hva den endelige løsningen ble.

4. Løsninger/alternativer

Kreftregisteret sin løsningsmodell innebærer en nettløsning med React, SurveyJS (Javascript klient) og Java (tjener), der kommunikasjonen foregår over HTTP. Fordelen med denne løsningen er at den gjør det mulig å bruke deler av løsningen andre steder, da klienten og tjeneren er to frittstående deler. Ettersom at Kreftregisteret driver med mye statistisk arbeid er det en fordel at denne løsning muliggjør å generere statistikk. For å sikre at den løsningen som er anbefalt er den riktige, har vi sett på andre teknologiske alternativer for tjener og for å lage skjema.

4.1. Frontend

Vi har valgt React da dette bygger på det allerede eksisterende systemet til Kreftregisteret.

For å gjøre koden gjenbrukbar og enkel å vedlikeholde trenger vi et rammeverk for å lage skjema, og har vurdert disse tre rammeverkene:

- SurveyJS
- Formik
- Formily

SurveyJS er et rammeverk laget av Devsoft Baltic. SurveyJS library er gratis og open /source. Devsoft tilbyr "SurveyJS Pro" med ekstratjenester som vi hverken skal betale for eller bruke. Grunnen til at vi har valgt SurveyJS er på grunn av hvor enkelt det er å skrive lettfattelig kode sammen med React, og at det er svært lett å utvikle betingete visninger av inputfelt. Man skriver alle spørsmålene, feltene og logikken i et JSON-objekt, og rendrer det i React (Devsoft Baltic OÜ, n.d.).

Formik er et rammeverk laget av Formium. Formik brukes av NASA, Nasdaq, Docker og flere store bedrifter. For å lage et skjema i Formik må man lage en "Form" på tilnærmet lik måte som i vanilla HTML5. Dette gjør at det blir mye boilerplate, og mer krevende å holde oversikt over betingelsene og logikken i skjemaet. Vi vurderer det slik at det kan bli krevende å vedlikeholde over tid på grunn av dette (Formium, n.d.).

Formily er et rammeverk laget av Alibaba. Formily sitt hovedfokus ligger i hvordan man kan lage logikk mellom de forskjellige feltene uten at ytelsen reduseres. Når man implementerer mange betingelser, kan det fort gå på bekostning av hastigheten. I følge Formily (2021), er det største problemet med rammeverket at det er svært vanskelig å sette seg inn i noe som vil bety at det krever mye ressurser og tid for å senere videreutvikle og vedlikeholde.

Kort oppsummert er fordelene og ulempene ved de tre rammeverkene slik:

Rammeverk	Fordeler	Ulemper
SurveyJS	• Skrives i JSON • Forholdsvis lett å vedlikeholde • Svært enkelt å binde felt med hverandre med betingelser • Lazy loading for store skjemaer • Kan lages i GUI	• Mindre brukt enn andre rammeverk • Krever at man skal lære seg et nytt rammeverk • Finnes rammeverk med bedre ytelse (Formily)
Formik	• Mye brukt i industrien • Stabilt • Ikke vanskelig å sette seg inn i	• Vanskeligere å binde mange felt sammen med betingelser • Dårlig ytelse
Formily	• Stabilt og brukt av Alibaba • Veldig god ytelse • Lett å binde felt sammen med betingelser • Man kan designe skjema i GUI	• Krever svært mye opplæring • Vanskelig å vedlikeholde med mindre man investerer mye tid på å lære seg rammeverket

4.2. Backend

I backend har vi valgt Java, da dette er brukt i de eksisterende systemene til Kreftregisteret.

Det finnes mange måter å utvikle webapplikasjoner i Java, blant annet Spring og Java EE. Java EE virker å være litt mer komplekst enn den andre nevnte, det trenger mer boilerplate kode og mer konfigurering. Spring Boot er en variant av Spring som har betydelig mindre konfigurering og boilerplate, men kommer med den ulempen at det kan være tregere (Brown, 2021). Siden Spring Boot er et rammeverk basert på Spring, er det enkelt å utvikle REST API og mikrotjenester.

I backend er det valgt Spring Boot fordi det skal være greit å skalere, er raskt å komme i gang med på grunn av Spring Boot sin “AutoConfigure”-funksjon og fordi Kreftregisteret har kompetanse på dette.

4.3. Valgt løsning

Løsningen som er valgt består av følgende teknologier:

- Java 17
- Spring Boot
- Hibernate
- React
- SurveyJS

Virkningen av å bruke et rammeverk over å bruke kun JavaScript for å utvikle klientsiden vil være at resultatet blir kode som er enkel å vedlikeholde. Virkningen av å bruke et annet programmeringsspråk, som for eksempel Python ville vært at utviklingstiden kunne blitt redusert da læringskurven på språket og utviklingen av REST API er enklere. Men fordi skalering av Python-

systemer ikke er like enkelt, i tillegg til at kompetansen hos oppdragsgiver ligger i Java, ville å velge Java hjelpe Kreftregisteret å integrere vår løsning i overordnede systemer hos Kreftregisteret.

5. Analyse av virkninger

Et nytt, moderne system har mange fordeler ved seg for Kreftregisteret. En liste over fordeler ved det nye systemet kan leses nedenfor:

- En modernisert versjon av tidligere løsning som er enkel å integrere med eksisterende systemer.
- Open-Source med fri lisens.
- Krever lite opplæring for brukeren.
- Uavhengig av avviklede InfoPath.
- Spesialtilpasset Kreftregisteret sine arbeidsoppgaver.
- Enkel å vedlikeholde og videreutvikle siden teknologiene som er brukt er Open-Source og er populær blant flere bedrifter og dermed utviklere.
- Bidrar til å effektivisere arbeidet de medisinske koderne gjør.

Noen ulemper med løsningen vil være:

- Må lære seg SurveyJS
- Tar tid, ressurser og opplæring for å videreutvikle løsningen
- Kan komme uforutsette kritiske feil som fører til nedetid og bruk av ressurser

6. Referanser

Brown, J. (2021, November 24). *Java EE vs. Spring: Which is more popular?* Xperti.Io. Hentet 23. Januar 2022, fra https://xperti.io/blogs/java-ee-vs-spring/#Java_EE_vs_Spring

Formium. (n.d.). *Overview | Formik*. Formik.Org. Hentet 23. Januar 2022, fra https://formik.org/docs/overview

Formily. (2021, November 24). *Competitive Product Comparison*. Formilyjs.Org. Hentet 24. Januar 2022, from https://formilyjs.org/guide#competitive-product-comparison