Digital akutjournal
Arkitekturdokument

Atoui, Josef Bäckman, Viktor Homssi, Rachel

Johansson, Tommy Jonsson, Jesper Lindgren, Felix

Runestam, Johan Wijk Stranius, Simon

8 maj 2020

Version 1.0

Status

Granskad	JR	2020-05-05
Godkänd	JR	2020-05-08

Versionshistorik

Version	Datum	Anmärkning
0.1	2020-02-23	Inlämning 1
0.2	2020-03-10	Inlämning 2
0.3	2020-04-20	Inlämning 3
1.0	2020-05-05	Justeringar och fixar av text och figurer inför inlämning 4.

Projektidentitet

Namn	Ansvar	Telefon	E-post
Josef Atoui	Teamledare	070-776 91 16	josat799@student.liu.se
Viktor Bäckman	Systemarkitekt	072-740 02 22	vikba308@student.liu.se
Rachel Homssi	Kvalitetssamordnare	070-487 53 23	racho401@student.liu.se
Tommy Johansson	Utvecklingsledare	072-224 86 94	tomjo891@student.liu.se
Jesper Jonsson	Konfigurationsansvarig	076-131 04 43	jesjo430@student.liu.se
Felix Lindgren	Testledare	070-875 14 23	felli675@student.liu.se
Johan Runestam	Dokumentansvarig	070-252 12 85	johru036@student.liu.se
Simon Wijk Stranius	Analysansvarig	073-909 59 14	simst932@student.liu.se

Innehåll

1 Systemöversikt

Syftet med VERA är att skapa ett journalsystem för akutvård. VERA ska hjälpa vårdpersonalen på en akutmottagning med det administrativa arbetet kring patienter. Från triagen, till undersökning och diagnostisering, till utskrivningen av patienten. VERA ska också hjälpa till med samarbete och organiseringen genom att ge vårdpersonalen en bra översikt över avdelningen.

Huvudmålgruppen och beställarna av VERA är Region Östergötlands olika akutvårdsmottagningar. De tänkta användarna är läkare, sjuksköterskor och undersköterskor som jobbar för Region Östergötland. De huvudkrav de har ställt på VERA är att det ska kunna ersätta deras befintliga pappersbaserade akutjournalsystem, möjliggöra parallellt arbete, förbättra översikten över avdelningen och patienter, och meddela vårdpersonalen om uppdateringar rörande patienter.

Följande är en lista med huvudkraven samt vilket nummer i kravspecifikationen de representerar.

1. I Vera ska det finnas olika vyer för patienter, avdelningsöversikt, ordinering av läkemedel, beställning av prover, teamöversikt och inmatning av journaler. Krav: 2.6.1, 2.6.2, 2.6.3, 2.6.5, 2.6.13, 2.6.15.
2. De ska gå att arbete parallellt med samma vy applikationen. Krav 2.6.2, 2.6.8.
3. Vera ska vara web-baserat och använda ramverket Angular. Krav 2.8.1, 2.8.2.
4. Journalen i Vera ska använda openEHR archetypes. Krav 2.8.3.
5. VERA ska kunna köras i Google Chrome. Krav 2.8.6.
6. Applikationen ska vara modulär. Krav 4.1.
7. Applikation ska kunna hantera 600 användare samtidigt. Krav 5.1.

2 Komponenter

I detta avsnitt beskrivs de olika komponenter som systemet kommer bestå av.

2.1 UI

UI-komponenten hanterar all interaktion med användaren. Detta inkluderar både att visa användaren information och att ta emot inmatning från användaren. Den agerar som ett mellanlager mellan användaren/applikationen. UI-komponenten har gränssnitten datorskärm, tangentbord och datormus.

2.2 Kommunikation

Kommunikationskomponenten hanterar kommunikation mellan applikationen och externa program. Detta för att säkerställa att kommunikationen är strukturerad och i rätt format. Detta kommer framförallt var kommunikation med EHRScape. Gränssnitten som komponenten kommer använda är ThinkEHR API [2].

2.3 Logik

Logikkomponenten hanterar den interna logiken i applikationen. Den styr logiken bakom applikationens funktioner och hanterar de cachade strukturer som finns. Den tar emot information från både UI-komponenten och kommunikationskomponenten samt serverar information till båda. Logikkomponenten kommer ha gränssnitt mot både UI-komponenten och Kommunikationskomponenten.

3 Struktur

I detta avsnitt beskrivs den struktur som utgör grunden till hur systemet ska byggas upp.

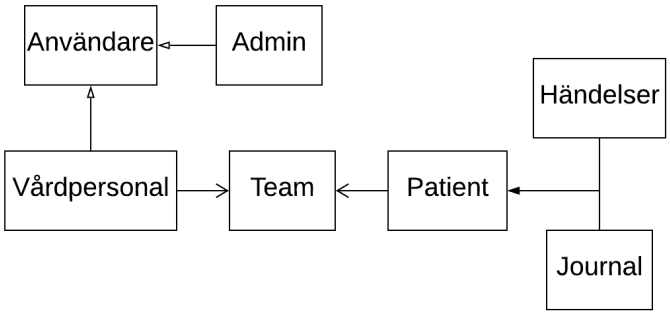
3.1 Statiska strukturen

Grunden i den statiska strukturen kommer vara objektet patient. Patient består av två delar. Del ett är en journal som innehåller all hälsoinformation om patienten. Del två är en händelse del som innehåller alla händelser relaterat till patienten. Detta kan vara information som att patienten har blivit undersökt av en läkare eller att ett prov har blivit beställt eller är klart.

Strukturen innehåller också klassen användare. Användare har två subklasser admin och vårdpersonal. Admin har möjligheter att administrera applikationen och applikationens användare. Vårdpersonal har möjligheten att använda applikationens olika produktivitets verktyg och funktioner. Detta inkluderar funktioner som att skapa patienter, uppdatera deras hälsodata och läsa patientjournaler.

För att strukturera vårdpersonal och patienter så används team. Team innehåller både vårdpersonal och patienter.

Se Figur 1 som innehåller ett diagram på den statiska strukturen.



Figur 1: Statisk Strukturdiagram

3.2 Dynamiska strukturen

Se Figur 2 som innehåller ett användningsfallsdiagram som visar systemets dynamiska struktur.

4 Antaganden och beroende

Denna del kommer ta upp tekniska kunskaper direkt relaterade till systemarkitekturen.

4.1 Kunskapsantaganden

- Angular och Typescript: Samtliga teammedlemmar förväntas ha god förståelse av Angular och Typescript.
- EHRScape och openEHR: Samtliga teammedlemmar ska ha goda kunskaper om hur EHRScapes API:et fungerar och strukturen av en openEHR-journal.
- CSS: En eller flera teammedlemmar ska ha goda kunskaper i standard CSS (Cascading Style Sheets).

4.2 Arkitektoniska antaganden

- Region Östergötlands IT-miljö kommer att ändras efter leverans.
- Arbetssättet på Region Östergötlands akutmottagningar kommer ändras efter leverans.

- Det kommer ske vidareutveckling av applikationen efter leverans.

4.3 Arkitektoniska beroenden

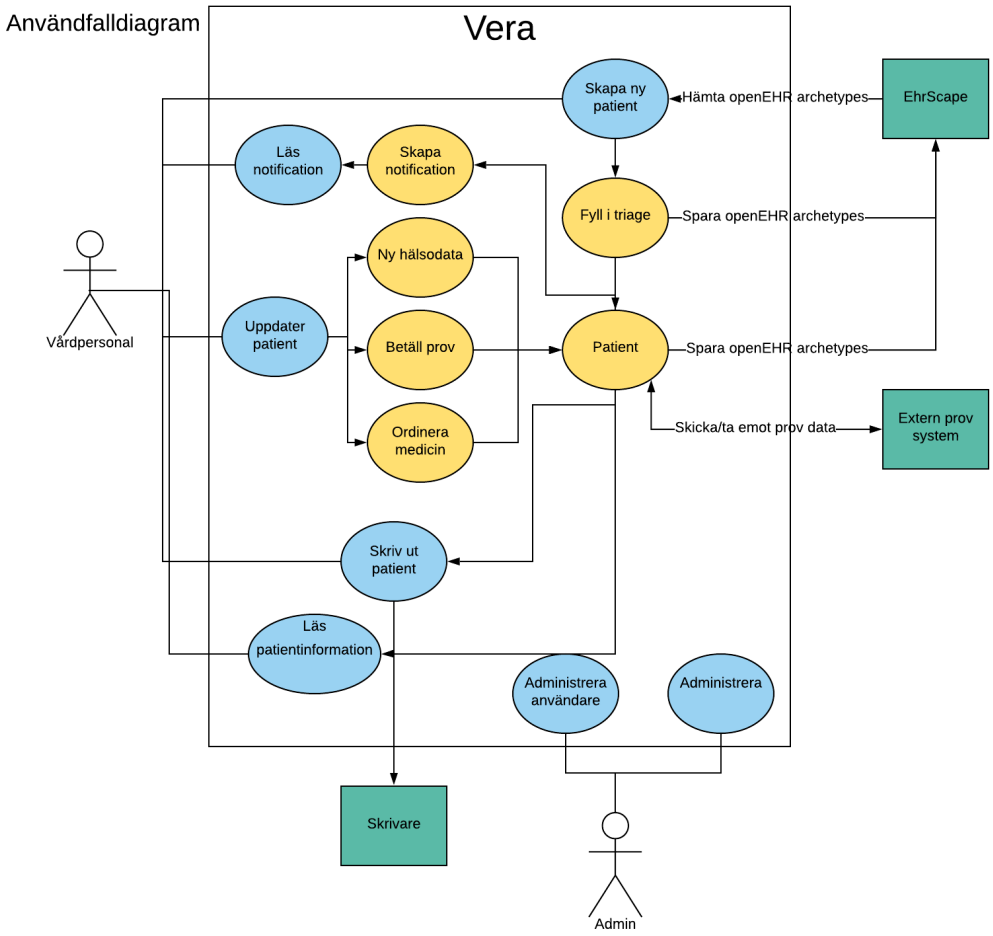
- EHRScape API.
- Google Chrome.
- Windows Server.
- openEHR-arketyper.

4.4 Tillgång till resurser

- För att säkerställa tillgängligheten av nödvändiga resurser för arkitekturen har ”open source” resurser samt resurser som Linköpings universitet tillhandahåller används så mycket som möjligt. Det har antagits att dessa kommer vara tillgängliga under projektperioden.
- Från Region Östergötlands IT-avdelning har teamet fått tillgång till EHRScape. Detta ger teamet möjlighet att på ett effektivt sätt jobba med openEHR-arketyper.

5 Beslut, begränsningar och motiveringar

- Applikationen ska byggas upp med ramverket Angular. Motivering är att Angular har den funkonalitet som kommer behövas för att bygga upp en komplett webbapplikation av hög kvalité. Det används av idag av Region Östergötland vilket möjliggör att vi kan få hjälp från dem. Det finns mycket resurser tillgängligt för utbildning och felsökning utöver Region Östergötlands kunskaper. Vilket gör att det passar väl för ett team som vårt där samtliga är nya till webbutveckling.
- Typescript och CSS ska användas för utvecklingen av applikationen. Motivering är att både Typescript och CSS fungerar väl med Angular och är rekommenderade att användas till Angular av [1].
- Applikationen ska köras i Azure. (Förkastat). Motivering var att hysa applikationen i Azure skulle hjälpa till med att göra leveransen mer komplett. Detta har valts bort då det krävde mycket arbete och medförde inte så mycket värde till slut resultatet. Att köra applikationen lokalt har valts istället.



Figur 2: Användningsfallsdiagram

6 Referenser

[1] *Angular.io*. https://angular.io/guide/typescript-configuration. Hämtad 2020-03-10.

[2] *ThinkEhr API*. https://www.ehrscape.com/api-explorer.html. Hämtad 2020-03-10.