Kandidatarbete

Redaktör: Pål Kastman

Datum: 2015-03-13

Version 0.1

Dokumenthistorik

Datum	Version	Utförda ändringar	Utförda av
2015-03-13	0.1	Första utkast av gemensam och individuella rapporter	Pål Kastman

Projektidentitet

Detta dokument gäller för grupp 7 i kursen TDDD77 på Linköpings universitet

Namn	Ansvarsområde	E-post
Daniel Rapp	Teamledare	danth407@student.liu.se
Daniel Falk	Analysansvarig	danfa519@student.liu.se
Jonas Andersson	Arkitekt	jonan111@student.liu.se
Albert Karlsson	Kvalitetssamordnare	albka735@student.liu.se
Erik Malmberg	Testledare	erima694@student.liu.se
Pål Kastman	Arkitekt	palka285@student.liu.se

Kund
Region Östergötland

Kundkontakt
Daniel Hall, daniel.hall@regionostergotland.se
Erik Sundvall, erik.sundvall@regionostergotland.se
Ingrid Hallander, ingrid.hallander@regionostergotland.se

Handledare
Lena Buffoni, lena.buffoni@liu.se

Handledare
Kristian Sandahl, kristian.sandahl@liu.se

Innehåll

Del I
Gemensamma erfarenheter och diskussion

1 Inledning

Detta avsnitt behandlar varför detta projekt utförs.

1.1 Motivering

Region Östergötland har idag ett system med handböcker som en sjuksköterska går igenom inför varje operation. I dessa handböcker finns bland annat förberedelse-uppgifter och plocklistor. Handböckerna är idag inte interaktiva på något sätt, istället skrivs plocklistan och förberedelseuppgifterna ut och bockas av för hand. Plattformen med handböcker kan heller inte återanvändas på olika avdelningar på grund utav licensproblem. Utöver detta system så finns ett annat separat system, som heter kartoteket, för uppgifter om vilka artiklar som finns och var i lagret de ligger. Detta gör att personalen som ska förbereda inför operationer behöver gå in i två olika system om de inte vet var alla artiklar ligger.

1.2 Syfte

Uppgiften som gruppen har fått är att skapa ett nytt system med handböcker som har interaktiva förberedelse-och plocklistor. Listorna ska uppdateras kontinuerligt när de bockas av så flera personer kan jobba på dem samtidigt. Plocklistan ska också innehålla uppgifter om var artiklarna ligger. Tanken är att personalen ska använda en iPad för listorna så de kan gå runt och plocka i lagret och bocka av samtidigt.
I mån av tid ska också extra funktionalitet implementeras. Till exempel sortera plocklistan med avseende på närmsta väg mellan artiklarna, lagersaldo och media i handböckerna.
Hela systemet ska ligga under en open-source licens så det kan användas fritt av alla.

1.3 Frågeställning

Rapporten ska besvara följande frågeställningar.

- Hur kan ett system för operationsförberedelser realiseras så arbetet blir lättare och mer effektivt?
- Kan man använda plattformen keystone för att bygga detta system?

1.4 Avgränsningar

2 Bakgrund

Studenterna som studerar kursen TDDD77 fick i januari 2015 ett uppdrag att utföra ett kandidatarbete. Först fick gruppen rangordna flera projektdirektiv för att sedan få ett av dessa uppdrag tilldelat sig. Grupp 7 fick då projektet operationsförberedelser som skickades in av Region Östergötland.

2.1 Wkhtmltopdf

Wkhtmltopdf är ett program som tar en hemsida eller en html-fil och skapar en pdf av den. Programmet har många olika parametrar som kan ändras t.ex. mediatyp så pdf-filen kan se ut som en utskrift av hemsidan. För att kunna använda detta programmet med nodejs finns det också en nodejs-modul med samma namn som skickar kommandon till programmet.

3 Teori

Node.js Keystone.js

4 Metod

I detta avsnitt beskrivs det hur arbetet med projektet har gått till.

4.1 Kravinsamling

Gruppen utgick ifrån projektdirektivet och började tänka över hur systemet skulle byggas. Det kom fram ganska snabbt att ingen i gruppen hade koll på hur operationsförberedelser går till, vilket gjorde att utbildning inom detta krävdes. För att få mer insyn så gjordes ett studiebesök på universitetssjukhuset i Linköping. Under detta studiebesök gjordes bestämdes också hur insamlingen av krav skulle gå till. Analysansvarig utsågs till ansvarig för detta. Denne skulle med hjälp av kunden arbeta fram en kravspecifikation som båda parter vara nöjda med. Detta gjordes genom flera möten och ett gemensamt dokument på Google drive där båda parter kunde gå in för att redigera och skriva kommentarer.

4.2 Utvecklingsmetod

Projektet har utvecklats iterativt med en utvecklingsmetodik som påminner mycket om SCRUM. Aktiviteterna i projektet har visats på en gemensam SCRUM-board med hjälp av webbsidan Trello. Boarden hade kategorierna TODO, DOING och DONE. När en medlem valt en aktivitet så märktes aktiviteten med medlemens namn och flyttades till den aktuella kategorin. Varje vecka har gruppen haft ett kort SCRUM-möte på 15 minuter. På mötet har varje gruppmedlem berättat vad som har gjorts den föregående veckan och vad som ska göras nästa vecka. SCRUM-mötet har oftast utförts i samband med varje veckas handledarmöte.

Ett system med alpha state cards har använts för att gruppen ska kunna veta hur långt projektet har fortskridit. Korten har även använts till att identifiera aspekter av projektet som kan förbättras och som gruppen behöver arbeta mer med. De aspekter av projektet som kunnat förbättras har använts som mål för kommande iterationer. De olika korten har uppdaterats kontinuerligt under projektets gång.

Nästan all utveckling har skett med hela gruppen samlad i samma lokal. Det har gjort att alla frågor om andra gruppmedlemmars kod har kunnat besvaratas snabbt. Det har gjort att arbetet kunnat hålla hög fart och att inga onödiga förseningar har uppstått på grund av osäkerheter i hur en del av koden fungerar. Problem har också kunnat lösas fort eftersom alla gruppmedlemmars olika kompetens har funnits tillgänglig.

4.3 Forskningsmetod

5 Resultat

5.1 Gruppens gemensamma erfarenheter

5.2 Översikt över de inviduella utredningarna

6 Diskussion

6.1 Resultat

6.2 Metod

6.3 Arbetat i ett vidare sammanhang

7 Slutsatser

8 Fortsatt arbete

Del II
Enskilda utredningar

A Kartoteket - Daniel Rapp

A.1 Inledning

Idag är information om Region Östergötlands operationsartiklar, så som priserna och placeringen i lagret på tandborstar, tandkräm, handskar och annan medicinsk utrustning, hanterat av ett internt system. Detta system kallas ett "*kartotek*".

A.1.1 Syfte

Det största problemet med det nuvarande kartoteket är att det inte är integrerat med systemet för att hantera handböcker.

I dagsläget, om en artikel slutas säljas eller Region Östergötland väljer att inte köpa in en viss artikel längre så tar de bort artikeln från kartoteket. Problemet som då uppstår är att detta inte reflekteras i handböckerna. Så om t.ex. en "*Oral-B Pro 600 CrossAction*" tandborste används i en "*Laparoskopisk sigmoideumresektion*", och tandborsten slutas säljas så tas den bort från kartoteket, men eftersom handboken för operationen inte är kopplad till kartoteket så uppdateras det inte att denna artikel inte längre finns i lagret.

Syftet med denna del av systemet är att lösa detta problem genom att integrera systemet som hanterar handböcker tillsammans med ett nytt kartotek, allesammans byggt på webben. När en artikel ändras eller tas bort i kartoteket så ändras den även i alla handböcker för operationer som kräver denna artikel.

A.1.2 Frågeställning

Frågeställningar:

- Kan man implementera ett kartotekssystem som uppfyller kundens önskemål?

A.1.3 Avgränsningar

Förutom ett förbättrat avcheckningsssytem så är Region Östergötland också i behov av ett bättre system för att hantera deras lager på ett mer automatiserat sätt. Bland annat så skulle de behöva ett system som låter dem checka in vilka varor från lagret de hämtat ut, istället för att checka av manuellt, vilket kan vara felbenäget.

Vi har dock valt att avgränsa oss från att bygga denna lösning, på grund av tidsbrister.

B Jonas Andersson

B.1 Inledning

Tidigare när jag har utvecklat webbprogram så har jag använt mig av PHP tillsammans med HTML, css och javascript. Jag har dessutom valt att skriva mycket själv och inte förlita mig på ramverk och bibliotek. I detta projekt valde jag, som arkitekt, istället att byta ut PHP mot node.js. Dessutom har jag lagt mycket vikt på använda bibliotek så mycket som möjligt för att slippa uppfinna hjulet på nytt.

B.1.1 Syfte

Syftet med denna del av rapporten är att analysera vad det finns för fördelar och nackdelar med att använda node.js gentemot andra vanliga språk för webben. Det ska även undersökas hur inlärningskurvan beror på språk och externa ramverk och bibliotek.

B.1.2 Frågeställning

Frågeställningar

- Vad finns det för fördelar/nackdelar med node.js gentemot andra språk för webben?
- Vad finns det för fördelar/nackdelar med att använda mycket externa bibliotek?

B.1.3 Avgränsningar

Det finns många programmeringsspråk som man kan använda i samma syfte som node.js. Eftersom jag enbart har tidigare erfarenheter av PHP och Python så kommer node.js jämföras mot dessa och inga andra.

B.2 Bakgrund

B.3 Teori

Node.js är en plattform för att skapa applikationer till framförallt webbservrar. Det finns en inbyggd pakethanterare vid namn npm som gör det enkelt att inkludera både små och stora bibliotek i sina projekt. Det är därför väldigt enkelt att använda sig av ett bibliotek istället för att skriva all funktionalitet själv.

Javascript är det programmeringsspråk som används i Node.js. På klienten, d.v.s. i webbläsaren, är man tvingad att använda sig av javascript eller något programmeringsspråk som kan kompileras till javascript. Genom att använda node.js får man därav samma språk på både server och klient.

B.4 Metod

B.5 Resultat

B.6 Diskussion

B.6.1 Resultat

B.6.2 Metod

B.7 Slutsatser

C Pål Kastman

C.1 Inledning

I detta projekt så har vi valt att inledningsvis skriva dokumentationen i Google Docs ordbehandlingsprogram för att i ett senare skede då det var dags att påbörja denna rapport gå över till att använda typsättningssystemet Latex.

Vi valde att göra på detta sätt för att så snabbt som möjligt komma igång, då man bland annat i Google Docs kan se live vad andra skriver.

C.1.1 Syfte

Syftet med denna individuella del är att undersöka funktionaliteten i Latex och väga fördelar mot nackdelar.

C.1.2 Frågeställning

- Vad finns det för begränsningar i Latex
- Kommer det att vara en fördel eller en nackdel för flera gruppmedlemmar att jobba samtidigt i samma delar av rapporten.

C.1.3 Avgränsningar

C.2 Bakgrund

C.3 Teori

C.4 Metod

C.5 Resultat

C.6 Diskussion

C.6.1 Resultat

C.6.2 Metod

C.7 Slutsatser

D Daniel Falk

D.1 Inledning

Jag har i detta projekt haft rollen som analysansvarig vilket innebär en analys av kundens behov och framställning av krav utifrån dessa. Rapporten beskriver hur kravframställningen gått till i detta projekt och hur prototyper har använts för att kommunicera idéer.

D.1.1 Syfte

Syftet med denna del av rapporten är att ta reda på hur man kan samla in krav som kunden har på produkten. Fokus ligger på användningen av prototyper med utgångspunkt i detta projekt.

D.1.2 Frågeställning

Hur kan man samla in krav som tillfredställer kundens behov? Hur kan man arbeta med prototyper för att framställa krav?

D.1.3 Avgränsningar

Rapporten har sin utgångspunkt i hur kravframställningen gått till i detta projekt. Metoden ska inte ses som ett allmänt tillvägagångssätt.

D.2 Bakgrund

Ett projekt faller ofta på grund av dålig kravframställning.

D.3 Teori

D.4 Metod

D.5 Resultat

D.6 Diskussion

D.6.1 Resultat

D.6.2 Metod

D.7 Slutsatser

E Automatiserade tester med Travis CI - Erik Malmberg

E.1 Inledning

Den här enskilda utredningen är en del av kandidatrapporten i kursen TDDD77 vid Linköpings universitet. Utredningen behandlar en del av utvecklingen av ett webb-baserat system för att underlätta förberedelser inför operationer på sjukhusen i Östergötland. Systemet utvecklades på uppdrag av Region Östergötland.

E.1.1 Syfte

Syftet med den här enskilda delen av kandidatarbetet är att ge insikt i hur kontinuerlig integration och automatiserade tester kan användas för att effektivisera testandet i ett projekt som använder en agil utvecklingsmetod. Speciellt ska det undersökas hur väl det går att använda Travis CI tillsammans med ramverket Jasmine.

E.1.2 Frågeställning

De frågeställningar som ska besvaras i den här enskilda delen av rapporten är:

- Hur kan man använda Travis CI tillsammans med Jasmine för att testa en webbapplikation byggd på javascript och node.js.
- Hur många tester hinner Travis CI köra på en sekund?
- Vilka typer av tester är svåra att utföra?

I svaret på den andra frågeställningen ska testfallen specifieras noggrant så att svaret inte blir tvetydigt.

E.1.3 Avgränsningar

Inga undersökningar kommer att utföras om hur andra lösningar än Travis CI kan användas för kontinuerlig integration. De testfall som kommer användas kommer uteslutande att vara skrivna med ramverket Jasmine.

E.2 Teori

Här beskrivs den teori som är nödvöndig för att förstå rapporten.

E.2.1 Vattenfallsmodellen

I vattenfallsmodellen genomförs all integration och alla tester efter att implementeringen är slutförd. Om ett problem då identifieras under integrationen så är det krångligt att gå tillbaka och åtgärda problemet. Det kan leda till förseningar av projektet.

E.2.2 Kontinuerlig integration och automatiserade tester

Kontinuerlig integration kan leda till att problemen identifieras tidigare i utvecklingsprocessen. Problemen blir då lättare att åtgärda. Enligt Karlsson (2014, 43) kan automatiserade tester effektivisera testprocessen och det finns många tillgängliga lösningar för att köra automatiserade tester. Några av de vanligaste är Travis CI, Codeship och Drone.

E.2.3 Travis CI

Travis CI är en webb-baserad tjänst för att köra automatiserade enhetstester och integrationstester på projekt som finns på GitHub. Travis CI är byggt på öppen källkod och är gratis att använda. Tjänsten har stöd för många olika programmeringsspråk, men det som är relevant för innehållet i den här rapporten är Javascript med Node.js. För att konfigurera Travis CI används filen .travis.yml som placeras i det aktuella projektets repository på GitHub.

E.2.4 Javascript

Javascript är ett programmeringsspråk som i första hand används på klientsidan på webbsidor. Javascript exekveras av webbläsaren och arbetar mot ett gränssnitt som heter Document Object Model.

E.2.5 Node.js

Node.js är en runtime environment för internetapplikationer. Det kan till exempel användas för att skapa webbservrar. Node.js är baserat på öppen källkod och det är enkelt att lägga till nya moduler för att anpassa det system man vill använda. För att lägga till nya moduler används node package manager (npm).

E.2.6 Jasmine

Jasmine är ett ramverk för testning av Javascript. Den node-modul som används är grunt-contrib-jasmine som använder task runnern Grunt för att köra testfall som skrivits med Jasmine. Grunt kunfigureras med filen Gruntfile.js.

E.3 Metod

Arbetet inleddes genom att Travis CI kopplades till projektets repository på GitHub. Kopplingen utfördes genom att administratören för repositoryn loggade in på travis-ci.org med sitt GitHub-konto och aktiverade en webhook för repositoryn.

De nödvändiga node-modulerna installerades med hjälp av node package manager (npm). Grunt installerades med kommandot: *npm install -g grunt-cli*. Grunt-contrib-jasmine installerades med kommandot: *npm install grunt-contrib-jasmine*.

E.4 Resultat

E.5 Diskussion

E.5.1 Resultat

E.5.2 Metod

E.6 Slutsatser

E.7 Referenser

Karlsson, Oskar. 2014. *Automatiserad testning av webbapplikationer*. Linköpings universitet. http://www.diva-portal.org/smash/get/diva2:727654/FULLTEXT01.pdf (Hämtad 15-04-19)

F Checkning av checklistor - Robin Andersson

F.1 Inledning

Vårt system ska innehålla olika typer av checklistor på olika webbsidor. Om flera användare är inne på samma sida samtidigt och en person checkar en checkruta så ska den checkrutan bli checkad för alla användare som är inne på den sidan.

Detta kommer att implementeras med hjälp av html, javascript och jquery samt Socket.IO för att checkrutor ska kunna uppdateras utan att webbsidan behöver uppdateras.

F.1.1 Syfte

Syftet med denna del av projektet är att flera sjuksköterskor samtidigt ska kunna plocka olika artiklar till en operation samtidigt utan att de råkar plocka samma artikel. Det ska även finnas en typ av checklista som innehåller olika förberedelser till en operation.

F.1.2 Frågeställning

- Går det att anpassa checklistan för en surfplatta medan den samtidigt innehåller information om var artiklar befinner sig samt hur många av varje artikel som behövs?
- Kommer Socket.IO vara tillräckligt snabbt för att flera personer ska kunna checka av artiklar samtidigt utan förvirring?

F.1.3 Avgränsningar

Eftersom denna del av projektet endast innehåller checkande av checklistor så saknas etiska aspekter.

F.2 Teori

Huvuddelen i implementeringen av checklistan är kommunikationen med Socket.IO som använder sig av websockets för att kommunicera mellan front-end och back-end. Information om Socket.IO finns på webbplatsen: *http://socket.io/*

F.3 Metod

Jag började med att fundera på hur kommunikationen skulle fungera på för sätt. Jag skissade ner olika förslag på ett papper och kom på det sättet fram till hur jag skulle implementera kommunikationen. Sedan implementerade jag den och fick den att fungera och efter det refaktoriserade jag koden för att få den kortare och mer lättläst.

F.4 Resultat

Jag kom fram till att när en användare går in på en operationsförberedelse så kommer denne in i ett rum. Varje gång en person sedan checkar en checkbox så skickas ett Socket.IO meddelande till servern som innehåller information om

vilken checkruta som ska checkas samt vilket rum checkboxen ska checkas i. Servern skickar sedan ett meddelande till det givna rummet vilken checkruta som ska checkas och alla klienter som är anslutna till det rummet checkar den givna checkrutan.

Figuren nedan visar detta flöde i ett sekvens liknande box-and-line-diagram.

När två klienter går in på en operation och en klient checkar en checkruta för första gången tar det strax under en sekund innan checkrutan checkas för den andra klienten. Därefter när någon klient checkar en checkruta så kan jag inte se någon fördröjning alls från det att en klient checkar en checkruta och en annan klient får den checkrutan checkad.

F.5 Diskussion

F.5.1 Resultat

Eftersom jag endast skickar data om vilken checkruta som ska checkas till de klienter som är inne på den operation som checkrutan blev checkad på så uppdateras checkningar snabbare än att göra den enkla lösningen att bara skicka datat till alla anslutna klienter. Att det tar nästan en sekund för en checkning att uppdateras på andra klienter för första gången är långsammare än förväntat. Men att det sedan går nästan helt utan fördröjning var mycket snabbare än förväntat.

F.5.2 Metod

Den metod jag använde mig av fungerade bra, men jag tror att jag skulle kunnat komma fram till samma resultat snabbare genom att göra kortare funktioner och vettigare namn redan från början istället för att göra något som funkar så snabbt som möjligt och sedan refaktorisera eftersom det blev väldigt förvirrande kod från början och jag var tvungen att sitta och tänka på vad kod jag skrivit gjorde. Men att skissa olika förslag på ett papper först tror jag var en väldigt bra idé.

F.6 Slutsatser

Den knappa sekundens fördröjning som är då en första checkruta checkas skulle kunna vara tillräckligt mycket för att ge en viss förvirring men eftersom det bara är en gång per operation så kommer detta antagligen inte vara ett praktiskt problem för sjuksköterskorna som ska använda systemet.

G Albert Karlsson

G.1 Inledning

G.1.1 Syfte

Syftet med denna enskilda utredningen är att undersöka olika kvalitetsaspekter då ett mindre projekt utförs av en liten grupp utvecklare.

G.1.2 Frågeställning

- Finns det några kvalitetshöjande processer som enkelt kan implementeras utan mycket resurser och kunskap?
- Behövs en kvalitetssamordnare i små grupper? Kan inte alla andra dela på detta arbetet?

G.1.3 Avgränsningar

Rapporten har sin utgångspunkt i hur kvalitetsarbetet har gått till i detta projekt och ska inte ses som ett allmänt tillvägagångsätt.

G.2 Bakgrund

I små projektgrupper blir kvalité ofta något som får stå åt sidan för andra saker som gruppen anser vara mer viktiga, så som implementering av nya funktioner. Det är ofta svårt att motivera utvecklare till att jobba med kvalitet eftersom arbete med kvalité sällan visar direkta resultat.

G.3 Teori

G.4 Metod

Mycket information kommer hämtas från andras erfarenheter inom kvalitetsarbete i små grupper. En sammanställning kommer sedan göras och det som många anser vara viktigt kommer sedan testas i projektgruppen.

G.5 Resultat

G.6 Diskussion

G.6.1 Resultat

G.6.2 Metod

G.7 Slutsatser