LoggiLue

Grupp 02

Verifiering och valideringsdokument

V. 1.0

2017-04-24

Dokumenthistorik

Datum	Version	Beskrivning	Författare
170406	0.1	Skrivande av coding guidelines, checklista för kodgranskning och processen för kodgranskning.	David
170407	0.2	Dokumentgranskning och riktlinjer för dessa.	Kajsa
170408	0.3	Slutförande av kodgranskning och hur denna process skall vara. Även infogande av en referens.	David
170410	0.4	Infogade referenser till dokumentgranskning, skapande av checklistor till dokumentgranskning och ett syfte för dokumentet har utformats.	Kajsa
170410	0.5	Användaranalys, beskrivning, infogandet av tabell och förklarning.	Oscar
170410	0.6	Testfall kravbaserad systemtestning samt Användbarhetstestning	Hannes
170410	0.7	Ordlistan införandet av nya ord och beskrivningar samt referenser till moscow modellen och användaranalys.	Oscar
170410	0.8	Beskrivning av syftet	Oscar, David
170410	0.9	Testprocess och mall för testrapporter	David
170410	1.0	Första slutförandet av version 1 av VoV-dokumentet	Hannes, Oscar, Kajsa, David
170421	1.1	La till en förklaring av Black-box testing och en referens.	Kajsa
170424	1.2	La till syfte för checklistor för kodgransknng. Formulerade stycket för kodgranskning mer objektivt. La till en förklaring av tabellerna för kravbaserad systemtestning. Ändring av stavfel från retrospekt 2.	David
170503	1.3	Borttagning av Cognitive Walkthrough metoden i användaranalysen. Modifierad text också. Bifogandet av användbarhetsanalysen v1.	Oscar

Innehåll

Verifiering och valideringsdokument

Syfte

Syftet med att skapa ett VoV-dokument är för att öka kvalitén och se till så man utvecklar en produkt som kunden vill ha. Även för att se till så att produkten byggs efter kravspecifikationen genom att få en överblick av hur kod- och dokumentgranskning ska gå till. Olika typer av tester kommer att genomföras för att säkerställa att detta uppfylls, däribland användbarhetstester för att se till så att användaren trivs med produkten men också olika kodrelaterade tester så att koden är så bra som möjligt. Dokumentet kommer även att innehålla checklistor för samtliga tester samt protokoll för dessa.

Ordlista

Black-box-testning	Man går igenom funktionerna på systemet utan att titta på koden.
Heuristisk utvärdering	Är en typ av analytisk utvärderingsmetod som kan användas utan att inblanda användare. Appliceras oftast på gränssnitt för att hitta diverse fel.
MoSCoW-modellen	En modell för att kunna prioritera krav.
Think Aloud användargränssnittet.	Verbalisera sina tankar när de rör sig genom

Referenser

[1] F. Tsui, O. Karam, B. Bernal,(2013) *Essentials of Software Engineering*, 3 upplaga, Burlington, MA 01803: Jones and Bartlett Publishers, Inc, S. 204, S. 220, S. 220.
[2] S. Eklund.(2010). *Arbeta i Projekt.* 3 upplaga. Studentlitteratur. Lund.
[3] Ingrid Domingues, Johan Berndtsson. Boken Användbarhet i praktiken http://anvandbarhet.se/start/
[4] M. Ted Boren and Judith Ramey. Thinking Alound: Reconciling Theory and Practice (2000) https://www.researchgate.net/publication/3230127_Thinking_aloud_Reconciling_theory_and_practice
[5] Nielsen, J. (1994b). Heuristic evaluation. In Nielsen, J., and Mack, R.L. (Eds.), *Usability Inspection Methods*, John Wiley & Sons, New York, NY.
[6] Chen & Macredie (2005). ”The assessment of usability of electronic shopping: A heuristic evaluation”, *International Journal of Information Management* 25, s. 516–532

Testprocess

I testningen ingår att designa testfall, köra programmet i en kontrollerad miljö och verifiera att outputen är korrekt. Testningen handlar om att ställa frågor till systemet och jämföra svaret med någon typ av ”korrekt” svar. Alla testfallen är baserade på vad som står i specifikationerna utan att kolla på koden. Testprocessen utgår ifrån olika testfall som blir grunden för hur man ska utföra testningen. De olika testfallen utgår från de olika kraven på systemet. Testfallen ska specificera hur systemet är tänkt att fungera och hur det ska hantera olika fel. Varje testfall har en unik id som identifierar just det testfallet.

Black-box testning [1] kommer utgöra den största delen av testningen. Black-box testningen innefattar tester utan att ha tillgång till koden och man testar bara systemets yttre mot en användare. Testningen kommer att genomföras av varje krav efterhand som kravet blir färdigt innan man börjar på nästa krav. Testningen skall inte utföras av författaren till koden utan av någon annan i gruppen. Dokumentationen av testningen kommer ske i ett separat dokument där de olika kraven som skall testas kommer att finnas, datum för testning, om kravet är uppfyllt eller inte, åtgärd ifall det inte är uppfyllt och vem som tar ansvar för att fixa det.

Granskning

Kodgranskning

Kodgranskning kommer att genomföras för att försöka hitta och minimera fel överlag i koden. Funktionen ”logga in” ligger som prioritet ett för granskningen. Detta för att den funktionen är mycket viktig för projektets funktionalitet och är en viktig grund för projektet. Koden kommer att granskas under ett möte och med hjälp av en checklista. Samtliga i gruppen kommer kodgranska, även författaren av koden.

Kodgranskningen kommer vara vad som kallas en ”inspection” [1] fast med vissa modifikationer för att bättre passa projektet. Nedan beskrivs gruppens process som kommer att användas för att granska koden.

- **Planering:** Planering för när mötet kommer att äga rum sker innan mötet. Samtliga gruppmedlemmar som skall delta informeras om detta. Personen som håller i kodgranskningen ser till att koden som skall granskas fungerar och skickar ut denna till deltagarna av mötet.
- **Förberedelser:** Alla deltagarna för kodgranskningen förväntas studera de filerna gruppen skall granska till innan mötet.
- **Möte:** Deltagarna har möte och går igenom koden tillsammans utefter checklistan som har skapats (se nedan). Alla genomför kodgranskningen, även författaren av koden. En person i mötet för ett protokoll och antecknar alla fel/misstag som kommer upp. Tiden ägnas bara till att hitta felen, inte att lösa de. Detta möte kommer att vara i två timmar och gruppen kommer granska filerna efter prioriteringen och se hur långt de hinner.
- **Omarbetning:** Personen som fick ansvar för att åtgärda något fel reparerar detta fel.
- **Uppföljning:** Någon i gruppen kontrollerar så att felet/felen är åtgärdade på ett korrekt sätt.

Riktlinjer för kod

Se separat dokument ”Coding_Guidelines_LoggiLue”

Checklista för granskningsmöte

Checklista kommer att användas för att förtydliga vad det är man skall kolla extra noga på under en kodgranskning.

Se separat dokument ”Checklista_Kodgranskning_LoggiLue”

Dokumentgranskning

Gruppen kommer att utgå ifrån en process som är utvecklad av Michael Fagan på IBM[1]. Processen följer sex olika steg och kräver en grupp på tre till sex personer som individuellt granskar en produkt och senare träffas på ett möte för att analysera arbetet mer i detalj. Eftersom att det är rekommenderat att vara tre till sex personer i granskningen kommer samtliga fyra medlemmar i gruppen att vara delaktiga under granskningen. Gruppen kommer göra en del ändringar i stegen för att anpassa dem till projektet. De listas i dem följande stegen:

1. Planering
Fagan skriver att en gruppmedlem, moderatorn, ser till så att dokumentet uppfyller en del krav. I gruppens fall kommer den dokumentansvarige att bli ansvarig för denna punkten och den kommer att se till så att dokumentet innehåller alla punkter som det ska och att alla är beskrivna. Mer utförlig granskning kommer ske senare av samtliga medlemmar under mötet.

2. Överblick
En överblick innebär, enligt Fagan, att man går igenom produkten och de relaterade områdena för att samtliga medlemmar ska bli uppdaterade och förstå vad dem handlar om innan mötet drar igång.
I gruppens fall kommer denna punkt inte att vara särskilt relevant eftersom samtliga medlemmar är med och skriver varje dokument. Även om det delas upp så att alla inte skriver allt tillsammans är alla medlemmar ändå medvetna om vad andras punkter handlar om. Detta eftersom gruppen, vid indelning av dokumenten, går igenom varje punkt och diskuterar lite innan den tilldelas. Ingen vet exakt vad som står där, men har ändå relativt bra koll på vad de innebär, därför känns en genomgång överflödig.

3. Förberedelser
Varje deltagare förväntas granska produkten och relaterad material inför mötet. Checklistor kan användas för att hjälpa till att identifiera eventuella problem.
Gruppen kommer inte att behöva anpassa denna punkten. Alla kommer att granska dokumenten innan mötet och checklistor för varje dokument finns.

4. Undersökning
Ett möte förbereds där produkten granskas. Den tilldelade läsaren presenterar arbetet (läsaren får inte lov att vara dokumentets författare) och de övriga fokuserar på att markera och dokumentera fel. Mötet går ut på att endast identifiera fel och går inte in på vart problemen uppstod eller hur dem ska lösas. Dessa sessionerna kommer att utspela sig under två timmar och efteråt kommer produkten antingen att accepteras som den är eller att ändras. Efter ändringarna sker antingen ett nytt möte och man börjar om på steg 1 igen, eller så kommer endast moderatorn att gå igenom och godkänna ändringarna.
Eftersom samtliga gruppmedlemmar är med och skriver dokumenten är alla författare av dokumentet. Detta gör så att tilldela en läsare inte är möjligt och läsaren kommer istället att bli moderatorn. I övrigt kommer denna punkt inte att behöva mer modifikation utan kommer att användas i sin helhet.

5. Omarbetning
Efter mötet ändrar författaren felen om det finns några. Författaren måste inte konsultera granskarna men får om den vill.
Återigen så finns ingen författare så moderatorn kommer att ändra felen.

6. Uppföljning
Ändringarna kontrolleras av moderatorn, eller så inspekteras produkten igen, beroende på resultatet.
Denna punkt fungerar bra i gruppens fall och den kommer inte att modifieras för att anpassas till gruppens behov.

Prioriteringen av dokumenten som görs kommer att prioriteras beroende på vad gruppen anser används mest i projektet. Dessa är projektplanen och kravdokumentet. Projektplanen anses ha högst prioritering eftersom det är den som agerar som en mall för projektet. Detta då den innefattar viktiga saker som planering, utvecklingsprocess och riskanalys som alla är viktiga under projektets gång. Kravdokumentets höga prioritering baseras på att kraven är väsentliga för projektets utformning. Genom att granska dessa dokument, se till att verkligen allt tas upp och är utvecklat utan flertydigheter, kommer dokumenten att fungera som en stadigare grund till hela projektet.

Riktlinjer för dokument

När det kommer till formaliafrågor som struktur, namngivning av filer, typsnitt, layout kommer dessa att följa mallen som är given för varje dokument och inget kommer att ändras därifrån.
Dokument som skrivs, där ingen specifik formalia angivits, kommer att skrivas med typsnittet ”Times New Roman” i storleken 12, standard radavstånd. Detta eftersom det är den formalia som är vanligast och ofta är den som är förbestämd för vissa dokument. Alla dokument ska även innehålla ett sidnummer (som placeras längst ner, centrerat på sidan) och ett sidhuvud som innehåller dokumentnamn, gruppnamn och gruppnummer. När det kommer till punkter där friare händer har givits, som kategorierna under funktionella och kvalitativa krav kommer dem att bestämmas gemensamt av gruppen. Det som man kom fram till var att under funktionella krav dela upp kraven i tre olika delar: krav som gäller konto, krav som gäller anteckningar och konto som gäller schema. Samtliga tre kategorier kommer i sin tur att vara uppdelade i två underkategorier: användarkrav och systemkrav. Detta anses vara relevant eftersom systemkraven är uppdelade mer tydligare krav som beskriver användarkraven. De krav som anses har en koppling till varandra kommer att markeras genom en indentering (ett klick på tab-knappen).

I projektplanen ska ett riskdiagram tas fram baserat på dem identifierade riskerna. Detta diagram kommer att göras i en typ av tabell med rutor som tydligt visar var på diagrammet risken befinner sig.

Checklista för granskningsmöte

Se separat dokument ”Checklista – Kravdokument” samt ”Checklista - Projektplan”

Testning

Kravbaserad systemtestning

Testning är en aktivitet som handlar om att utvärdera systemets kvalité, och att förbättra den, genom att identifiera defekter och problem. Bästa sättet att uppnå kvalité är genom att ha en väl-definierad process som passar organisationen och projektet bäst [1]. En högkvalitativ process leder till större chanser till högkvalitativa system. Den kravbaserade systemtestningen utgår ifrån kraven och går igenom dessa en efter en enligt projektplanens prioritering. Mest tid kommer läggas på de kraven som är av högsta prioritet.

Prioritering

Testfallen kommer att prioriteras på samma sätt som kraven enligt MoSCoW-modellen[2]. Testerna är kopplade till kravspecifikationen och utgår från kraven.

Användbarhetstestning: ”Think Aloud”

Användbarhetstestning är en metod inom programvarutestning för att utvärdera ett system eller en tjänst genom att testa den med personer som är tänkta att använda produkten [1]. Metoden är en av de centrala metoderna inom användbarhet.

Användbarhetstestningen kommer innehålla en intervju där testpersonen kommer gå igenom de olika testfallen och funktionerna i systemet. Testpersonen kommer få olika uppgifter och bli dirigerad att utföra olika funktioner. Till detta kommer det utformas en intervju/uppgiftsmall som kommer innehålla de olika uppgifterna för testningen. Alla testpersoner kommer därför ställas inför samma uppgifter för att se om deras sätt att lösa problemen skiljer sig ifrån varandra. De olika uppgifterna kommer vara kopplade till de viktigaste av kraven och gå igenom den viktigaste funktionaliteten [3].

Metoden som kommer användas under intervjun är ”Think-Aloud” [4] eller ”Talk-Aloud protocol” vilket innebär att personen som utför testet kommer bli tillbed att tala ut sina tankar som kommer upp under testet. Testpersonen kommer till varje testfall få förklara det den ser och hur den tänker för att ta sig igenom de olika uppgifterna. I ”Think-Aloud” testning ber man testpersonerna att använda systemet och kontinuerligt tänka högt, det vill säga att helt enkelt verbalisera sina tankar när de rör sig genom användargränssnittet[4].

Användarhetstestning kommer att genomföras när helheten av designen är klar för att få användaren att förstå hur systemet i helhet kommer se ut och fungera.

Prioritering

De olika testerna kommer att prioriteras på samma sätt som kraven enligt MosSCoW-modellen.[2] De viktigaste funktionerna kommer främst att involveras i testerna och testpersonen kommer spendera längst tid på att undersöka dessa. De viktigaste kraven kommer få mest tid i användbarhetstestningen.

Användbarhetsanalys

Användbarhetsanalys: Heuristisk utvärdering

Heuristisk utvärdering är en välkänd och etablerad metod som används flitigt inom olika användarbarhetstester men också inom diverse ämnesgrupper som Interaktionsdesign och Informationsarkitektur. Den heuristiska utvärderingen utgår från olika principer som har utformats av en stor mängd forskare, Nilsens[5] principer anses vara dem främsta och mest kända samt mest använda av utvärderare inom användbarhet. Dessa 10 principer [6] kommer ligga som grund till den heuristiska utvärderingen(förklarningarna är tagna från *Wikipedia*):

Nr	Namn	Förklarning
1	*Visibility of system status*	Gränssnittet ska hålla användaren informerad om det bakomliggande systemets tillstånd.
2	*Match between system and the real world*	Gränssnittet ska efterlikna verkligheten och beskriva den på ett språk användaren förstår.
3	*User control and freedom*	Användaren ska ha möjlighet att utveckla egna strategier samt ångra saker som går fel.
4	*Consistency and standards*	Gränssnittet ska följa konventionerna för den plattform det körs på och användaren ska inte behöva undra vad olika formuleringar betyder i olika situationer.
5	*Error prevention*	Gränssnittet ska vara uppbyggt så att det förebygger fel.
6	*Recognition rather than recall*	Användaren ska inte behöva komma ihåg var olika funktioner ligger i gränssnittet utan bör känna igen dessa enkelt.
7	*Flexibility and efficiency of use*	Användaren ska kunna anpassa gränssnittet till sig själv och sina arbetsuppgifter.
8	*Aesthetic and minimalist design*	Gränssnittet ska inte presentera irrelevant information då varje irrelevant informationsbit konkurrerar med relevant information.
9	*Help users recognise, diagnose and recover from errors*	Felmeddelanden ska vara enkelt formulerade, förklara vad som gick fel och föreslå en lösning.
10	*Help and documentation*	Information för att hjälpa användaren ska vara lätt att hitta i, fokusera på användarens arbetsuppgifter, visa konkreta steg som ska utföras och inte vara för omfattande.

Metoden kommer anpassas till projektet genom att identifiera de viktigaste funktionerna i LoggiLue's system och hur dessa samspelar med gränssnittet. Gränssnittet ska vara där för att stödja användaren när denne utför dem olika funktionerna i systemet.

Poängen med analysen är att utvärdera gränssnittet och försöka göra en objektiv bedömning för att identifiera potentiella hinder och problem men också att belysa de goda aspekterna med designen. Vidare ska utvärderaren försöka lägga fram lösningar för dessa så att gränssnittet blir så användarvänligt som möjligt vilket i sin tur leder till att användaren kan nyttja alla funktionerna på ett enkelt sätt.

Testfall kravbaserad systemtestning

Tabellerna nedan visar de olika testfallen gruppen tar fram från kraven av produkten. Varje testfall har ett unikt ID som identifierar just det testfallet, en rubrik, diverse förberedelser som skall göras innan testet, teststeg som görs under testet och ett förväntat resultat av testet.

Globala felhanteringar

ID	T1
Rubrik	Server tillgänglig
Förberedelser	När servern inte kan nås/ inte tillgänglig. Servern returnerar ett 500 fel.
Teststeg	1. Försöka komma in på webbsidan 2. Komma in på de olika sidorna
Förväntat resultat	Ett meddelande som säger att servern inte kan nås och att användaren ska försöka igen senare.

Logga in/ut

ID	T2
Rubrik	Logga in
Förberedelser	På startsidan och ska logga in.
Teststeg	1. Användaren är redan registrerad och använder sina uppgifter för att logga in.
Förväntat resultat	Användaren kommer in på systemet och kan se respektive programs schema.

ID	T3
Rubrik	Logga ut
Förberedelser	Inloggad på systemet
Teststeg	1. Användaren ska logga ut
Förväntat resultat	Användaren lämnar sin systemet och hamnar på startsidan vid inloggningen.

ID	T4
Rubrik	Felhantering vid inloggning
Förberedelser	Fel uppgifter vid inloggning
Teststeg	1. Användare uppger fel användarnamn och/eller lösenord.
Förväntat resultat	Användaren stannar på inloggningssidan och ett meddelande som säger att användaren har fyllt i fel uppgifter i systemet.

Registrering

ID	T5
Rubrik	Registrera användare

Förberedelser	Användare inte registrerad, och går till registreringssidan.
Teststeg	1. Användaren fyller i sina uppgifter på registreringssidan. 2. Användaren missar att fylla i någon uppgift.
Förväntat resultat	Uppgifterna registreras i databasen och användaren får ett meddelande att den har blivit registrerad på sidan. Återgår till inloggningssidan. 2. Användaren får ett meddelande att den missat att fylla i vissa uppgifter. (eller) Knappen kommer inte gå att trycka på.

ID	T6
Rubrik	Felhantering registrering
Förberedelser	Användare inte registrerad, och går till registreringssidan.
Teststeg	1. Användaren fyller i användarnamn som redan finns. 2. Användaren fyller i e-mail som redan finns. 3. Användaren fyller i två olika lösenord.
Förväntat resultat	Användaren får ett meddelande om att någon av dessa uppgifter redan finns i systemet. 3. Användaren får ett meddelande att lösenorden inte matchar.

Anteckningar

ID	T7
Rubrik	Lägga till en anteckning
Förberedelser	Inloggad i systemet
Teststeg	1. Välj föreläsning 2. Ange anteckningen 3. Klicka på ”ladda upp”
Förväntat resultat	Meddelande om att anteckning blivit korrekt uppladdad. Användaren återvänder till sidan för respektive föreläsning.

Schema

ID	T8
Rubrik	Se schema från Kronox
Förberedelser	Inloggad på systemet
Teststeg	1. Inloggad på systemet 2. Befinner sig på startsidan

Förväntat resultat	Användaren skall kunna se sitt schema från sin utbildning på hemsidan.

ID	T9
Rubrik	Felhantering för Kronox schemat
Förberedelser	Inloggad på systemet.
Teststeg	1. Kronox API är inte tillgängligt 2. Kronox schema är inte nere
Förväntat resultat	Felmeddelande att Kronox schemat just nu inte är tillgängligt, vänligen försök igen.

Spårningsmatris

Exempel test 2 går igenom kravet FAK-2

Inloggning

Testfall/krav	FAK-2	FAK-3	FAK-4
T2	X		
T3		X	
T4	X	X	X

Registrering

Testfall/krav	FAK-1
T5	X
T6	X

Anteckningar

Testfall/krav	FAA-1	FAA-2
T7	X	X

Schema

Testfall/krav	FAS-1	FAS-2	FAS-3
T8	X	X	X
T9	X	X	X

Granskningsprotokoll

Granskningsprotokoll Kodgranskning KG1

Se bifogat dokument ”KG1 – Protokoll för kodgranskning”.

Granskningsprotokoll Dokumentgranskning DG1

Se bifogat dokument ”DG1 – Kravdokument - protokoll”.

Granskningsprotokoll Dokumentgranskning DG2

Se bifogat dokument ”DG2 – Projektplan- protokoll”.

Testrapporter

Testrapport kravbaserad systemtestning ID: TKS1

Se bifogat dokument ”Testrapport_kravbaserad_systemtestning”

Analysrapporter

Användbarhetsanalys

Se bifogat dokument ”Användbarhetsanalys_V1”