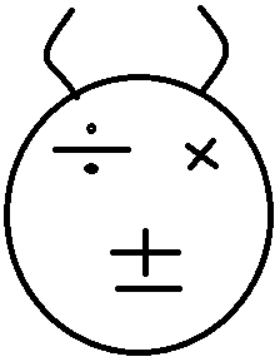
Kvalitetsplan

PUM09

2024-04-12

Version 3.0



Status

Granskad	ReviewerName	2015-xx-xx
Godkänd	ApproverName	2015-xx-xx

Projektidentitet

Grupp E-post: emiho191@student.liu.se

Hemsida: http://www.liu.se/grouppage

Beställare: Kristian Sandahl, Linköpings universitet
Tfn: 013-28 19 57
E-post: kristian.sandahl@liu.se

Kund: Jörgen Blomvall, Linköpings universitet
Tfn: 013-28 14 06
E-post: jorgen.blomvall@liu.se

Handledare: Lena Buffoni
Tfn: 013-28 40 46
E-post: lena.buffoni@liu.se

Kursansvarig: Kristian Sandahl, Linköpings universitet
Tfn: 013-28 19 57
E-post: kristian.sandahl@liu.se

Projektdeltagare

Namn	**Roll**	**E-post**
Mabest Amin (MA)	Arkitekt	mabam091@student.liu.se
Odin Dahlström (OD)	Utvecklingsledare, Vice Databasansvarig	odida723@student.liu.se
Emil Holmstedt (EH)	Teamledare	emiho191@student.liu.se
Martin Hultgren (MH)	Kvalitetssamordnare, Databasansvarig	marhu242@student.liu.se
Casper Erik Nerf Kanefall (CK)	Testledare	casne582@student.liu.se
Eric Van Nunen (EVN)	Analysansvarig	eriva185@student.liu.se
Wiliam Puranen (WP)	Konfigurationsansvarig	wilpu732@student.liu.se
Yadgar Suleiman (YS)	Dokumentansvarig	yadsu309@student.liu.se

INNEHÅLL

emiho191@student.liu.se

DOKUMENTHISTORIK

Version	Datum	Utförda förändringar	Utförda av	Granskad
1.0	2024-02-17	Första utkast	MH	YS, EH
2.0	2023-03-05	Åtgärdade kommentarer från handledare, var god se kommentarerna eller issue #57 för specifika delar. Inkluderat Feature Branches"	MH	EH
2.1	2023-03-25	Lade in Git CI i kvalitetsbestämmelser produkt samt relevant begrepp. Uppdaterade vilka formatörer och standarder som används till kod.	MH	
2.2	2023-04-12	Lade till de ändringar som gruppen har identifierat än så länge	MH	
3.0	2023-04-12	Inlämning 3	MH	YS

1 INTRODUKTION

1.1 Syfte

Kvalitetsplanen existerar för att alla aktörer skall lita på att kvaliteten hålls över spannet av projektet. Detta i process, produkt samt relevant dokumentation.

1.2 Omfattning

Systemet som denna kvalitetsplan omfattar benämns hädanefter som produkten. I detta dokument kan produkten refereras uppdelad i två distinkta delar. Applikationsdelen benämns hädanefter som *applikationen* eller *appen* och databasdelen benämns hädanefter som *databasen*.

Produkten är överlämnad från tidigare projekt vilket att innebär att vissa kvalitetsåtgärder är ärvda. De ärvda kvalitetsåtgärderna är kod- och kommentarstandarder.

2 BEGREPP OCH FÖRKORTNINGAR

- **Scrum** - En agil metod att utveckla ett projekt.
- **Scrum board** - En visuell metod att hantera uppgifter där stadiet av uppgifterna hamnar i olika kategorier.
- **Standup** - Ett sorts möte där medlemmarna snabbt förmedlar till reseten av gruppen hur arbetet går. Namnet kommer ifrån att medlemmarna oftast står så att mötet går snabbt.
- **Backlog** - En ordnad levande lista av allt arbete som bör göras.
- **Menti** - Ett web-baserat verktyg för att samla in åsikter anonymt.
- **Dart** - Ett programmeringspråk utvecklat av Google för att utveckla olika typer av applikationer.
- **Effective Dart** - En stilguide för hur dartkod skall utformas.
- **Flutter** - Ett ramverk för att utveckla mobil-, web- och datorapplikationer med samma kodbas.
- **IDE** - Integrated developer environment.
- **Merge** - En process där två grenar i ett versionshanteringsverktg sammanfogas.
- **Merge request** - En granskningsprocess där en annan utvecklare granskar ändringar innan de läggs in i projektet.
- **Git CI** - En process där förbestämda procedurer körs då en merge request skapas och ändras.

3 ÖVERSIKT

3.1 Parter

Kunden till projektet som beskrivs i detta dokument är Jörgen Blomvall, tillhörande institutionen för ekonomisk och industriell utveckling (IEI) på Linköpings universitet. Producenten är grupp-09 i kursen TDDD96, Kandidatprojekt

i programvaruutveckling, på Linköpings universitet vårterminen 2024. Andra parter kopplade till projektet är Lena Buffoni (IDA), handledare för projektgruppen, samt Kristian Sandahl (IDA), examinator i kursen TDDD96, Kandidatprojekt i programvaruutveckling.

Alla medlemmar i grupp-09 har individuellt ansvar att följa kvalitetsplanen. Ansvar att kvalitetsplanen kan hållas samt att den uppfyller kraven faller på kvalitetssamordnare.

3.2 Kvalitetskrav

Det finns ett flertal krav för produkten beskrivna i kravspecifikationen, nedan beskrivs de krav som omfattar kvalitet:

Krav	Version	Beskrivning	Prioritet
1	1.0	Det skall max ta 150ms mellan att en uppgift är avklarad tills att nästa uppgift visas på skärmen	Hög
2	1.0	Koden skall ha en *test coverage* på minst 80%	Hög
3	1.0	Applikationens *textkontrastförhållande* ska ha ett värde på åtminstone 4.5:1 på liten text och 3:1 på stor text.	Hög

3.3 Verktyg

Här beskrivs de verktyg som används under utvecklingen av produkten samt hur de används.

3.3.1 *Discord*

Discord är ett kommunikationsverktyg för röst-, video- och textsamtal. Den primära anledningen Discord används i projektet är för att separera kanaler och ämnen in i olika kanaler.

3.3.2 *Gitlab*

Gitlab är ett verktyg för versions- och projekthantering. Förutom versionshantering används Gitlab i detta projekt även för att hantera ärenden genom Gitlab Issues.

3.3.3 *Gitlab Issues*

Gitlab Issues är ett verktyg för att planera, hantera och organisera olika ärenden. Detta görs genom ett flertal funktioner i samma verktyg där de viktigaste för gruppen är:

- Möjligheten att kategorisera uppgifter.
- Möjligheten att tilldela uppgifter till personer.
- Möjligheten att sätta tidsgränser.
- Möjligheten att länka till uppgifter som relaterar, blockerar eller kan blockeras.

3.3.4 *Dart Analyzer*

Dart Analyzer är Darts version av kodanalys för att hjälpa utvecklare följa rätt kodstandard. De flesta IDEs har detta inbyggt med hjälp av en flutter modul.

3.4 Standarder

Dokumentet följer standarden för IEEE Software Quality Assurance Plans [1]

4 KVALITETSÅTGÄRDER

4.1 Produktåtgärder

Här beskrivs de åtgärder som införs för att hålla kvaliteten på produkten hög.

4.1.1 *Kodstandard*

Alla filer skall ha ett tydligt och fokuserat ämnesområde för att undvika gigantiska filer. Utöver detta skall all kod i filen vara uppdelad i funktioner eller metoder med ett tydligt syfte. Nya variabler, funktioner, metoder samt klasser skall vara kända förkortningar eller engelska ord som följer programmeringsspråkets riktlinjer.

Pythonkod skall följa stilguiden PEP8 och samtliga gruppmedlemmar skall använda en linter kallad Ruff [1].

Dartkod skall följa stilguiden Effective Dart [2]. Samtliga gruppmedlemmar skall använda Dart-Analyzer som kommer med flutter.

4.1.2 *Kommentarstandard*

Kommentarer krävs för funktioner samt metoder där ett eller flera av följande kriterier är uppfyllda:

- Funktionen används utanför filen där den är definierad.
- Metoden används utanför klassen där den är definerad.
- Funktionen eller metoden är ej av trivial storlek.
- Funktionen eller metoden innehåller ej trivial logik.

Kodgranskaren har sista ordet när det gäller mängden kommentarer.

Både stilen samt innehållet i kommentarerna bestäms av respektive språks stilguide. För dartkod se Effective Dart [3] och för pythonkod se avsnittet för kommentarer i stilguiden PEP8 [4]

1 https://github.com/astral-sh/ruff
2 https://dart.dev/effective-dart
3 https://dart.dev/effective-dart
4 https://peps.python.org/pep-0008/

4.1.3 *Kodgranskning*

Varje *merge* till huvudgrenen skall granskas av en annan person som ej är författare. Förslagsvis är detta en annan person i arbetsgruppen. Om en merge inte har tagits upp av en medlem inom 48 timmar bör den tilldelas av mergens ansvariga. När en medlem har blivit tilldelad en merge så skall den vara igenomläst inom 48-72 timmar beroende på storlek och importans. Vid denna kodgranskning skall granskaren se över att:

- Programmet startar.
- Programmet klarar av att köra de ändrade och relaterade delarna.
- Alla ändringar följer relevant kodstandard.
- Alla ändringar följer relevant kommentarstandard.

Vid ej godkänd granskning så måste en *merge* revideras med kommentarer från granskaren. Då granskningen accepteras är det upp till arbetslaget att sammanfoga. Detta ser till att koden i någon mån alltid fungerar samtidigt som det lättar underhållet av koden för framtiden.

4.1.4 *Git CI*

För att säkerställa att produkten alltid fungerar och håller gruppens standard på kodkvalitet används *Git CI*. De processer som körs när en ny *merge request* skapas är följande:

- Applikationen byggs till en Android APK för att säkerställa att applikationen är körbar.
- Backend sätts upp i en docker container för att se till att det går samt inför testning.
- Statisk kodanalys körs på frontend och/eller backend beroende på vilken del som har ändrats. Denna kodanalys verifierar att utvecklaren har följt gruppens standard på kod samt formaterar om ifall någon del är felformaterad.
- Alla frontend tester körs med flutters inbyggda system av testing.
- Alla backend tester körs mot docker containern.

4.1.5 *Kodtester*

Kodtester är en viktig del av att säkerställa kvaliteten hos produkten. Gruppens strategi för tester beskrivs därför i ett annat dokument kallat testplan [2].

4.1.6 *Användartester*

Användartester skall utföras i samarbete med kund i April. Specifikt datum diskuteras fortfarande.

4.2 Processåtgärder

Här beskrivs de åtgärder som implementerats för att hålla kvaliteten på processer.

4.2.1 *Roller*

Alla medlemmar i gruppen har åtminstone ett område de ansvarar för, detta ger en klar person att vända sig till när det gäller överblickande frågor. Dessa roller visas i tabellen ovan sidhuvudet kallad projektdeltagare.

4.2.2 *Krav och spårbarhet*

Kraven på projektet har många olika stilar och sätt att verifieras, för många för att tas upp i detta dokument. För att spåra vilka krav som är uppfyllda eller ej används excel.

4.2.3 *Gitlab Issues*

Gitlab Issues har ett flertal olika delar. Nedan representeras riktlinjerna för dessa delar i punktform.

- All text i issues skall vara på svenska.
- Alla issues som estimeras av skaparen att vara större än femton timmars arbete skall brytas ner till mindre delar.
- Alla issues som har varit på brädet i mer än två veckor brytas ner till mindre delar.
- Beskrivningen i issues skall följa den förinställda mallen i Gitlab. Detta inkluderar krav, acceptans, beroende med mera. De delar i mallen som ej är relevanta kan lämnas tomma eller tas bort.
- Länkade issues skall fyllas i.

4.2.4 *Feature Branches*

Under utveckling skall alla implementationer i koden använda en egen ny *branch* beroende på ämne. Denna branch har ett flertal regler:

- En branch skall bara existera en maximal längd av en sprint.
- En branch skall följa namnkonventionen av ett issue nummer följt av ett beskrivande namn på engelska.
- En branch skall beseras på minst en issue.
- En branch skall implementera alla ändringar i huvudgrenen innan en merge request skapas.

4.2.5 *Merge request*

Alla *merge requests* skall skrivas på svenska samt följa den förinställda mallen i Gitlab. De delar i mallen som ej är relevanta kan lämnas tomma eller tas bort.

4.2.6 *Commits*

Alla Commits skall skrivas på engelska och skall ha en beskrivning passande storleken. Varje gång individen stänger ner utvecklingsmiljön eller är klar för dagen skall en commit skapas, utöver detta är det upp till individen.

4.2.7 *Riskhantering*

Riskhantering berörs i projektplanen [3].

4.2.8 *Utbildning*

Alla i gruppen har eget ansvar att utbilda sig om grundliga ämnen. Större ämnen bör tas upp på veckomötet för att se till att stora undersökningar ej sker två gånger.

Om det är en större implementation som resten av gruppen behöver ta del av skall det bokas in en studiegrupp. Nedan kommer en kort sammanfattning av vilka studiegrupper som har genomförts, vad de handlade om samt när de genomfördes.

Tabell 2: En tabell över vilka studiegrupper som har genomförts

Namn	**Sammanfattning**	**Datum**
Developer	Utvecklingsmiljön demonstrerades samt sattes upp för alla i gruppen	2024-02-22
CI	Gruppens formaterings- och testverktyg demonstrerades tillsammans med en guide på vilka rutiner gruppen har för en merge request	2024-03-04
Testing	Diskussioner om hur gruppen skall testa samt vad gruppen skall testa. Även exempel togs upp	2024-04-03

4.2.9 *Veckomöte*

Veckomötet skall användas för att evaluera den tidigare veckan samt planera den kommande samtidigt som det skall identifiera problem som kan vara i vägen under sprintens gång. Under veckomötet skall gruppen diskutera:

- Vad som gjorts.
- Vad som skall göras.
- Vad som blockerar eller kan blockera folks nuvarande och kommande arbete.
- Vilken hjälp som behövs.
- Vad som har fungerat mindra bra i gruppen samt i arbetet.
- Vad som har fungerat bra i gruppen samt i arbetet.

Utöver detta bör gruppen diskutera övriga frågor som uppstått under veckan.

4.2.10 *Arbetsfördelning*

Gruppen skall delas upp in i arbetsgrupper sprintvis vid varje veckomöte, teamledaren har sista ordet om indelning. Dessa grupper ska bestå av två eller fler gruppmedlemmar. Detta innebär inte parprogrammering utan är menat att vara en säkerhetslina vid oväntade situationer och ger en tydlig grupp att diskutera frågor med.

4.2.11 *Kommunikation*

För att kommunicera skall gruppen använda Discord för diskussioner och Messenger för meddelanden som behöver nå ut snabbt. På vardagar skall alla i gruppen ta del av informationen från de gemensamma kanalerna inom tjugofyra timmar. Detta låter gruppen ha en informationskälla där information kan sparas under längre tid samt en källa för snabb information. Discord har valts över andra kommunikationskanaler med liknande funktionalitet på grund av gruppens tidigare bekantskap med verktyget.

4.3 Dokumentåtgärder

4.3.1 *Versioner*

Heltalsdelen av versionsnumret på ett dokument som versionhanteras skall ökas inför varje inlämning för att visa att stora ändringar har skett. Decimaldelen av versionsnumret på ett dokument som versionhanteras skall ökas vid vid ändringar som ej är triviala. Dessa ändringar skall också dokumenteras i dokumentets dokumenthistorik.

4.3.2 *Granskning*

Alla dokument som ej är av trivial storlek eller trivial betydelse behöver granskas av minst en person i gruppen innan inlämning. De punkter en granskare skall se över är följande.

- Besvaras alla frågor som skall besvaras?
- Är språket korrekt?
- Är källhänvisningen korrekt?
- Har texten en röd tråd?
- Är tabellen för versionshantering korrekt och uppdaterad?

5 PROCESS I FOKUS

5.1 Scrum

Nedan beskrivs ramverket scrum samt gruppens anpassning av det. Utöver det beskrivs mätpunkter, utvärdering och hur processen skall förbättras.

5.1.1 *Beskrivning*

Scrum är ett agilt ramverk som etablerar en grundlig struktur för att lösa komplicerade problem. Idén bakom scrum är att iterativt och responsivt arbeta fram till en färdig produkt där delprodukten alltid kan användas under utveckling. Detta görs genom att dela upp projektet i tidsluckor som kallas sprints. För varje sprint sätts ett klart mål upp som hela gruppen skall jobba mot. Detta resulterar i att delar av slutprodukten implementeras del för del. Dessa sprints utvärderas i slutet med vad som kallas en scrum retrospective där gruppen diskuterar bättre samt sämre punkter och ändrar hur de arbetar för nästa sprint. Denna uppdelning, målsättning och utvärdering leds av en Scrum Master. [4]

Praktiskt sätt fungerar scrum på följande sätt. Allt arbete som gruppen behöver göra för att slutföra produkten lever i en levande ordnad lista som kallas backlog. När en sprint börjar väljer Scrum Mastern ett mål för sprinten. När det är gjort tar Scrum Mastern de punkter som är relevant till målet ifrån gruppens backlog till scrum tavlan för sprinten. När en utvecklare sedan jobbar på en punkt tas den ifrån **att göra** till **pågående**. Då utvecklaren är klar med en punk tas den över till **tester och granskning** där tester och granskning utförs enligt gruppens överenskommelser.

5.1.2 *Gruppens anpassning*

Scrum är utvecklat för att lösa större och komplicerade problem. Då gruppens projekt handlar om vidareutveckling av en applikation där problemen är mindre anpassningar behöver vissa delar av ramverket ändras till behov. Sättet gruppen kommer använda scrum på är följande.

- Längden av sprints är två veckor. Då projektet är över en väldigt begränsad tid så ger detta gruppen möjligheten att snabbt ändra hur de jobbar för att inte slösa tid.
- Ett mål per arbetsgrupp och sprint istället för ett mål för hela gruppen per sprint. Då projektet kräver ett flertal mindre ändringar är ett ensamt mål väldigt begränsande. Detta ger gruppen enkelheten med få mål samtidigt som det ger flexibiliteten att jobba på flera uppgifter samtidigt.
- Restrospectives i två delar istället för en. Då gruppen som helhet är nya till arbetsmetoden bestämdes det att utvärdera sprinten både i mitten samt på slutet i formen av ett veckomöte, se kapitel 4.2.9 för mer information. Detta låter gruppen planera och fundera på ändringarna till nästa sprint samt identifiera problem tidigare.
- Teamledaren agerar som Scrum Master.

Målet med dessa åtgärder är att ramverket skall kunna användas till kraven som gruppen har samt att ramverket ska kunna anpassas snabbt och effektivt då det behövs.

5.2 Mätning och Utvärdering

Nedan beskrivs de mätningar som skall genomföras under projektets gång. Dessa mätningar skall användas för att bestämma ändringar i gruppens processer samt användas som underlag för utvärdering.

Första mätpunkten bestämmer hur bra gruppen har bedömt arbetsupplägget. Detta görs genom att mäta antalet issues som är kvar eller inte kvar på tavlan efter en sprint. Målet är att jobba igenom alla issues som är uppsatta för sprinten precis innan den är färdig. Denna mätpunkt utvärderas från följande två frågeställningar:

- Hur många issues blev över efter sprinten som gruppen inte han färdigt med?
- Arbetade gruppen igenom issues innan sprinten var färdig och isåfall hur långt innan?

En minskning i dessa representerar en bättre estimering av tid och arbetsuppgifter. Vilket representerar en förbättring av arbetsprocessen. En ökning i dessa visar att gruppens estimation av tid och arbetsuppgifter blir sämre. I båda fallen behövs en analys göras för att ta nytta respektive förhindra anledningen till resultatet.

(Notera att denna mätpunkt inte har gett oss någon relevant data hittills så den bör bytas ut inom snar framtid)

5.3 Ändringar

Nedan beskrivs alla ändringar som har genomförts i gruppens arbetssätt. Då gruppen (hittils) inte har identifierat något stort bristande problem är kapitlet uppsatt i ett flertal mindre ändringar. Alla ändringar har en ändringsbeskrivning, orsak, förhoppning och utvärdering.

5.3.1 *Sprintutvärdering*

Ändringsbeskrivning: Sprintutvärderingen hanteras genom anonyma beskrivningar på *menti* istället för direkt diskussion i mötet.

Orsak: Gruppmedlemmar upplevde att gruppen ej fick tillräckligt med återkoppling runt brister för att utvärdera samt förbättra gruppens processer.

Förhoppning: Detta bör underlätta insamlingen av information genom att låta personer som är mindre bekväma med att ta upp problem i större grupper upplysa observerade brister.

Utvärdering: (NOTE: Delvis utvärdering) I detta fall är det svårt att avgöra om återkoppling gruppen nu har samlat in beror på möjligheten att vara anonym men det är klar kontrast i mängden feedback som gruppen fick efter denna ändring etablerades. Nedan visas antal unika positiva samt negativa åsikter insamlat än så länge.

Innan menti (6 utvärderingar)	4
Efter menti (1 utvärdering)	10

Tabell 3: Antal unika punkter återkoppling insamlat innan och efter menti användes

5.3.2 *Kommunikation - Discord*

Ändringsbeskrivning: Det implementerades ett antal nya kanaler i gruppens discord för att främja bättre kommunikation och möjligheter till hjälp. Dessa nya kanaler var:

- Hjälp: En kanal för att fråga om all möjlig tänkbar hjälp.
- Bokningar: En kanal för att kommunicera bokningar av rum för att enklare kunna arbeta tillsammans och diskutera frågor.
- Arbetslagskanaler: En kanal per arbetslag istället för strikt frontend och backend kanaler. Detta låter hela gruppen se diskussioner inom ett område istället för att det hanteras i direktmeddelanden.

Orsak: Under en menti fångades en stor grupp återkoppling som tillsammans visade på vissa brister. Nedan är ett urval av alla återkopplingar som mottagits:

1. Det är en tydlig struktur på vem som skall göra vad.
2. Medlemmar dålig uppfattning av vad andra medlemmar har gjort.
3. Inte fått den hjälp som behövs.
4. Gruppens timmar är låga.

Då återkoppling 1 och 2 är delvis motsägande så evaluerades detta till att vara ett problem ihopkopplat med återkoppling 3. Efter diskussion tolkades dessa återkopplingar som att det inte fanns en klar och tydlig väg att fråga om hjälp.

Förhoppning: Detta bör förenkla möjligheten för alla i gruppen att få hjälp, både med utvecklingen samt de begränsade timmarna som gruppen har.

Utvärdering: TBD

5.3.3 *Kommunikation - Standup*

Ändringsbeskrivning: Varje dag som hela gruppen inte möts på förmiddagen skall alla skriva en kort sammanfattning av vad de gjort dagen innan samt vad de skall göra den dagen. Detta skall skrivas i kanalen i discord som kallas *Standup*.

Orsak: Anledningen till denna ändring var densamma som i förändringen ovan kap 5.3.2. De mer specifika är punkterna är 2, att medlemmar inte riktigt har en uppfattning av vad andra gör och 4, att gruppens timmar lagda är låga.

Förhoppning: Det finns ett flertal resultat som gruppen önskar att detta medför. Dessa är följande:

- Bättre insikt vad medlemmar gör för samtliga.
- Tidigare uppfångst av dålig arbetsfördelning
- Förhöjd känsla av ansvar och personlig motivation.

Utvärdering: TBD

REFERENSER

[1] "Ieee standard for software quality assurance processes," *IEEE Std 730-2014 (Revision of IEEE Std 730-2002)*, pp. 1–138, 2014.

[2] PUM09, "Testplan optimalt lärande," 2024.

[3] ——, "Projektplan optimalt lärande," 2024.

[4] K. Schwaber and J. Sutherland, "The scrum guide," https://scrumguides.org/scrum-guide.html, 2020, [Online; refererad februari 15, 2024].