Software Quality Assurance

**MoonEyes: Case Management System for Metro Detective Agency**

Senior Design I Summer 2023

Team Members:

Kylie Callison

Erika Valle-Baird

Kendall Gesek

Pamela Shahu

Table of Contents

[**1.0 Introduction 3**](#_ghw88196icu3)

[1.1 Scope and intent of SQA activities 3](#_1j4au8pm3q4)

[1.2 SQA organizational role 3](#_xiqnma16y3yp)

[**2.0 SQA Tasks 4**](#_m2isk440837a)

[2.1 Task Overview 4](#_6s3ox4757mzp)

[2.1.1 Description of SQA task m 4](#_ki6xl7vkyk20)

[2.1.2 Work products and documentation 5](#_2o29vifqr78t)

[2.2 Standards, Practices and Conventions (SPC) 6](#_xcr8nfcxgj0s)

[**3.0 Reviews and Audits 8**](#_w8r1m7xzbk54)

[3.1 Generic Review Guidelines 8](#_qjxid7bg5xb3)

[3.1.1 Conducting a Review 8](#_voma3yzbcaih)

[3.1.2 Roles and Responsibilities 9](#_h10mcbjgk6a5)

[3.1.3 Review work products 9](#_n5dtkyowx665)

[3.2 Formal Technical Reviews 10](#_87uo6kjeoj04)

[3.2.1 System specification review 10](#_6x1wy7vo226w)

[3.2.2 Software project plan review 11](#_ozem3vnxat38)

[3.2.3 RMMM review 11](#_zd3uxb8cqc14)

[3.2.4 Requirements reviews (models, specification) 12](#_6harpd3tcq5m)

[3.2.5 Data design review 13](#_22gsfkudcmrs)

[3.2.6 Architectural design review 13](#_v7a6k469vjgx)

[3.2.7 Interface (GUI) design review 14](#_jaqhhdz5bpgb)

[3.2.8 Component design review(s) 15](#_ksrmn2u1xf00)

[3.2.9 Code Reviews 15](#_l0rncc894ipg)

[3.2.10 Test specification review 16](#_der4svoegdn)

[3.2.11 Change control reviews and audits 16](#_wo8itk2esn37)

[3.3 SQA Audits 17](#_ihhvgjh0wyeu)

[**4.0 Problem Reporting and Corrective Action/Follow-up 18**](#_whwxw956o7wn)

[4.1 Reporting mechanisms 18](#_x3h21s679c4b)

[4.2 Responsibilities 18](#_p98j9zfconjz)

[4.3 Data collection and evaluation 19](#_29j7t24pfdia)

[4.4 Statistical SQA 21](#_rfhhgf5na2f2)

[**5.0 Software Process Improvement Activities 22**](#_p3d06as810xi)

[5.1 Goal and objectives of SPI 22](#_igi9x7hsa6v9)

[5.2 SPI tasks and responsibilities 23](#_s6m7ipdp5lf3)

[**6.0 Software Configuration Management Overview 23**](#_g1sfwprq8ktw)

[**7.0 SQA Tools, Techniques, Methods 24**](#_8vpu4l7gmjiv)

[**8.0 Appendix 30**](#_h49zxih2oiw)

[8.1 Traceability Matrix 30](#_yrmbfzg770f9)

[8.2 References 30](#_26hxf05pm7aw)

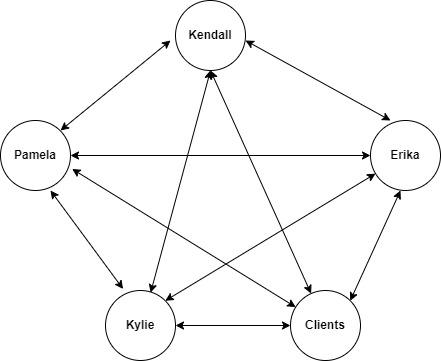
# 1.0 Introduction

## 1.1 Scope and intent of SQA activities

The objectives of the SQA team are to ensure project management processes by the development of quality outputs. The SQA team will provide quality assurance by adhering to client design specifications. To guarantee the product meets customer specifications, the SQA team has set up frequent meetings with clients for verification and planning. This was done in an effort to prevent risks that could affect the quality of the final software. Walking through each stage of development with our clients safeguards our software against risks that affect production quality and maintains quality control by sustaining a customer feedback loop. The SQA team sets out to reach its goals of customer satisfaction and defect elimination by monitoring and improving any weaknesses identified in processes. For our team to verify our software is meeting its requirements, adhering to predefined standards, and uncover errors pertaining to function, logic, and implementation, we will be conducting FTRs. FTRs will be conducted via walkthroughs and technical reviews. Due to the size of our team, each team member will employ multi-dimensional roles and will all remain members of the SQA team.

## 1.2 SQA organizational role

Due to the size of our team, each team member will employ multi-dimensional roles and will all remain members of the SQA team. We will employ an egoless team structure in which all members will be involved in decisions and cross reference all work.



# 2.0 SQA Tasks

## 2.1 Task Overview

### 2.1.1 Description of SQA task *m*

**Description of SQA Task 1**

The team will check the Software Requirements Specification throughout the back-end coding process to ensure the product is following the original design. The team will also check the document on a weekly basis to ensure the code and the product adheres to the standards set forth in the document. This check will look like the team member referring to the Software Requirements Specification document for the specific element they are working on that week or are developing. This will ensure that the software is meeting the quality and expectations of the client and working as intended.

**Description of SQA Task 2**

The team will refer to the Software Requirements Specification when developing or working on the user interface to verify the product is following the intended design. The team will utilize the Software Project Plan as well as the Requirement Specification document when it comes to developing specific elements of the user interface. The documents will be referenced on at least a weekly basis to ensure that the user interface is functioning properly and aligned with the client’s expectations of how it should look and function.

**Description of SQA Task 3**

All team members will test and evaluate the user interface. The criteria used will be a heuristic approach to reflect the ease and efficiency of use, overall attractiveness, intuitiveness, error prevention, and consistency with different windows. For our client, the user interface is extremely important, so it is crucial that it is evaluated frequently and honestly so the product can be attractive and easy to understand.

**Description of SQA Task 4**

Each member of the team will evaluate and test the back end of the product to ensure the database and cloud capabilities are working as designed. The criteria for these evaluations will include logic errors, cybersecurity checks, database management, automated testing framework, and network availability and scalability. This will ensure the product is free of defects, performing efficiently, and satisfactory for the client.

**Description of SQA Task 5**

There will be specific team members who will lead the SQA process. They will manage the SQA reviews, control the flow of information to the rest of the team, and lead the SQA meetings. The team members will review all the change requests and assign a level of priority to each of them for implementation. The rest of the team will be included in the discussion on how to prioritize each of the change requests, so all team members are on the same page with the implementation of the requests.

### 2.1.2 Work products and documentation

**Work products and documentation for Task 1**

If there are any major changes requested or found, there will be a separate change log which will be accessible to all team members. This log will be up to date with documentation for each completed change request in date order, so all information is available to team members so other elements of the project can be adjusted accordingly.

**Work products and documentation for Task 2**

Any major changes will be documented in the change log with proper documentation referring to the change. This log is accessible to every team member with all reported change requests, so other elements of the project can reflect any changes made.

**Work products and documentation for Task 3**

Any suggestions and concerns will be expressed in team meetings. A change request must be filed for any changes to occur to the user interface as mentioned in Task 2. The team members responsible for the user interface will make the necessary adjustments to ensure the product is satisfactory for the client.

**Work products and documentation for Task 4**

Weekly team meetings will allow for any changes or concerns to be discussed among all team members. Any changes will be documented in the change request log by date order and be accessible to all team members at any time. The team members will adjust the system to ensure the product is performing at its best level.

**Work products and documentation for Task 5**

During each SQA review, all information will be recorded for future use. All change requests will have an assigned ranking for implementation. This ranking will be developed in a team meeting with all team members since we have an egoless structure. The change request will also be added to a log with all information, ranking, and the date the item was reviewed by the team.

## 2.2 Standards, Practices and Conventions (SPC)

Close Contact with Client

Due to client request inconsistencies, close contact with the client will allow our team to improve product quality with frequent meetings discussing all information pertinent to the project. This will allow for almost immediate clarification for any questions our team may have while working on the project. This will also allow for our client to obtain and try our working prototype sooner, allowing for faster turnaround times for correcting any necessary issues. This will both save our team time along with saving our client costs as they will have a working product sooner.

Any changes will undergo our Software Change Management process, as described in the software project plan. A change is initiated with change request identification, both issue resolution and impact summary are required to determine a plan of attack and how this change may affect our system. From this point a change request is logged, to ensure proper documentation. Our team then must review and evaluate such a change, deciding if we will move forward with the change or if the change is not viable. If the change will be beneficial to our program, we then implement the change and update our log. If the change is denied, a revision can be made to then reassess the proposed change as a team at a later time.

**2.3 SQA Resources**

There will be a variety of resources used to structure the SQA process such as meetings to specifically review any change request and activities to control and guide the process. There will be team members who will focus on the SQA process and can communicate the SQA information to the rest of the development team. These people will supervise the entire quality control process to ensure the product is aligned with the requirements set in the Software Requirements Specification document. They will also be responsible for designating the priority ranking of every change request reported, with a higher priority given to those functional defects compared to aesthetic ones. This ranking will determine which changes need to be implemented first by the team. Every change request is to be reviewed by the SQA leaders before being shown to the rest of the team. Since all changes requested need to be implemented by the rest of the team due to size, they will also oversee the quality analysis of the product. The software developers are responsible for developing a product that adheres to the Software Requirements Specification document. This means before and after each change request is completed, the developers will ensure the change is within the bounds of the original product. Because of the team’s size, each member will frequently be testing the current software product for any defects or enhancements to ensure the product is functioning properly and as intended. There will also be audits done frequently to support the proper functioning of the product. Each team member will have access to the source code to assist with the debugging process, this will also help in the early development stages to catch defects before a change request is needed. At this time, there will be no additional software or hardware to assist in the quality assurance process, but it may be requested later by the clients as the business and team could continue to grow. As of now, the team will use a log to track all change requests and create proper documentation for each one. This will be organized by date of implementation to easily track the changes and will be accessible for every team member for review.

# 3.0 Reviews and Audits

*Software Engineering: A Practitioner's Approach* describes the primary objective of a formal technical review as the following, “to find errors before they are passed on to another software engineering activity or released to an end user.”. Additionally, it serves as a means to ensure the software is developed in a uniform manner and meets its requirements. The text also mentions that the focus of a formal technical review, or FTR, is on the work product.

## 3.1 Generic Review Guidelines

Generic guidelines for a review are outlined below. Details for conducting a review, the roles and responsibilities of those on the review team and the review work products are described in the following sections.

### 3.1.1 Conducting a Review

General guidelines for conducting a formal technical review are outlined below.

See [3.1.2 Roles and Responsibilities](#_h10mcbjgk6a5) for a description of each role mentioned.

After the completion of the work product by the producer, the review will be scheduled by the review leader. For our purposes at least 3 people must attend the review. The leader will also create an agenda for the review and facilitate the meeting.

Once the review begins the producer will walk through and explain their work product/function/code. The reviewers will listen carefully and ask questions and point out any errors during the walk-through. One of the reviewers, the recorder, will also take note throughout the presentation and discussion and also go through the checklist. The review leader will be responsible for making sure everyone has an appropriate tone and also limit any longer debates. Once the review is complete the reviewers will make one of three decisions.

1. Accept the product fully
2. Accept the product with small easy modifications
3. Reject the product to severe errors (must wait for another review)

After a decision has been reached the meeting will end. The recorder will also create a small review summary report detailing the meeting and decision.

Note that due to the length of the Moon Eyes project, these types of FTR’s will be for larger work products such as: the creation of a database and the appearance of user interfaces, etc, more are described below in [3.2 Formal Technical Reviews.](#_87uo6kjeoj04)

For smaller work products and functionalities, a more informal review can be conducted with 2-3 people and a checklist can be developed and used to ensure software quality.

### 3.1.2 Roles and Responsibilities

Because we have a relatively small team of four, in each formal technical review that takes place we may need to have people cover multiple roles. In many cases, every member of the team will be present during a formal technical review. The text recommends about three to five people should be involved in a technical review. As mentioned earlier we have an egoless team structure which applies to the SQA processes. We have decided on the following roles for our reviews and their responsibilities. Additionally, because the review team also makes up our entire development team, it is likely that there will be scenarios where a single person has multiple roles. As mentioned above, we will have multidimensional roles.

* **Producer (1):**

The producer is the one who created the product that is being reviewed and is responsible for presenting the product. During the review meeting they will “walk-through” the work product and explain.

* **Reviewers (2):**

The reviewers will be responsible for listening to the producers and gently pointing out errors and raising questions.

* + **Recorder (1):**

The recorder is one of the reviewers who will generate notes during the review meeting. Each time an error is raised, it will be recorded. Additionally, they will be responsible for creating the review summary report after the meeting. They will also have the same responsibilities as the reviewer role.

* **Review Leader (1):**

Schedules and establishes an agenda for FTR. The review leader conducts/leads the meeting. Ensure proper attitude is maintained and limit longer debates on an issue. They will also have the same responsibilities as the reviewer role.

### 3.1.3 Review work products

Documents, forms, lists produced as a consequence of the FTR are described below.

**Notes**

The recorder will take notes during each meeting about all products and conversation that takes place during the FTR. These will include any issues raised and also a reviews issue list is produced.

**Review Summary Report**

From this, a “Review summary report” will be created and shared after each review meeting. This is a short report that details, what was reviewed, who was it reviewed by and what was the end result of the review. Any actions or changes that need to be taken will also be listed here as well. The notes will be attached to the report as well.

**Review Checklist**

Simple review checklists will be developed beforehand and are necessary for formal and informal reviews.

Additional work products will be mentioned in the *3.2.n.3* work products section of each formal technical review.

## 3.2 Formal Technical Reviews

The details for formal technical reviews of eleven total items can be found below.

### 3.2.1 System specification review

**3.2.1.1 Description and focus of the review**

The review will focus on the quality, accuracy and effectiveness of the system specification we outlined. All system requirements will be evaluated. If no separate system specification is developed for Moon Eyes, the SRS will be reviewed and is explained more thoroughly in [3.2.4 Requirements reviews (models, specification)](#_6harpd3tcq5m)

**3.2.1.2 Timing of the review**

The system specification review will be held shortly after the completion of the SyRS. This review will take place before the software is developed in order to ensure all requirements and system details have been reviewed for any issues to save time later in the development.

**3.2.1.3 Work products produced**

The recorder will take notes on the system specification and create a review issues list and will also utilize the checklist. A review summary will also be made to address any changes that will be made and the formal decision on the acceptance of the system specification.

**3.2.1.4 Review System Specification checklist**

* Does the system specification align with the Moon Eyes scope?
* Does the system specification clearly break up each component into subcomponents?
* Is there anything that needs to be added to the system specification?

### 3.2.2 Software project plan review

**3.2.2.1 Description and focus of the review**

The review focused on the project plan document that was created. The main emphasis will be on reviewing the time estimates and schedule for the development of Moon Eyes. The review will also ensure that all resources needed are mentioned in the plan.

**3.2.2.2 Timing of the review**

This review will be held before or during the first couple weeks of development, after the completion of the software project plan. This is because we need to ensure that we have a good basis for the schedule in order to track software development progress.

**3.2.2.3 Work products produced**

The recorder will take notes during the review and also ensure that all questions have been addressed on the checklist. Additionally the recorder will create a review summary with any changes to the project plan like cost estimates, schedule or software/hardware resources needed.

**3.2.2.4 Review Software project plan checklist**

* Has too much or too little time been assigned for any specific component in Moon Eyes?
* Have tasks been delegated equally between each team member to each of their abilities?
* Has all resource needs been addressed?
* Is there enough overall development time to complete all previously established requirements?

### 3.2.3 RMMM review

**3.2.3.1 Description and focus of the review**

The RMMM review will focus on the risks outlined in the risk mitigation, monitoring and management plan. The main emphasis will be on making sure all risks are covered and that the risk management actions outlined can appropriately deal with the risks.

**3.2.3.2 Timing of the review**

This review will be during the first couple weeks of development, after the completion of the RMMM. This is because the plan needs to be established in case of risk realization and also allow us to make decisions based on the risk implications.

**3.2.3.3 Work products produced**

The recorder will take notes on all issues raised such as risks that were missed, additionally they will make sure everything on the checklist was addressed. Also a review summary will be produced with final decision and any changes that need to be made to RMMM.

**3.2.3.4 Review RMMM checklist**

* Are risks categorized in terms of impact and likeliness to occur?
* Is there a clear outline of steps to be taken once a risk is realized?
* Have all project risks been considered?
* Has a specific person been assigned to deal with each risk mitigation action?
* Is the RMMM compatible with our time and resource constraints?

### 3.2.4 Requirements reviews (models, specification)

**3.2.4.1 Description and focus of the review**

This review will be focused on the list of requirements and deliverables related to the software requirements specification document. The main goal is to evaluate the requirements and see if they are clear and extensive enough.

**3.2.4.2 Timing of the review**

This review will also occur in the initial stages of project development after the creation of the document. This is to ensure that we have a solid foundation before beginning development by having all requirements clearly stated and addressed. This is very important because the requirements document will hold the development team accountable.

**3.2.4.3 Work products produced**

The recorder will take notes throughout the interview and note any issues raised with requirements. Again the checklist will be followed along with a review summary at the end that will detail which requirements need to be changed.

**3.2.4.4 Review Requirements review checklist**

* Are all functional and non-functional requirements addressed in the document?
* Are the requirements possible to do in the given timeframe?
* Does each requirement align with the stated project scope?
* Has the requirements addressed every request of the clients?

### 

### 3.2.5 Data design review

**3.2.5.1 Description and focus of the review**

This review's main focus will be on ensuring our data design for Moon Eyes is organized and concise. We will be evaluating the data model and making sure it serves all the functional requirements and is as efficient and also structured correctly as possible.

**3.2.5.2 Timing of the review**

This review will also need to occur early on and design and can be revisited later in the project as well if addition and modifications to the data design are necessary in a later prototype. But generally this first review needs to be done early so backend development can begin.

**3.2.5.3 Work products produced**

Again, notes will be taken by the recorder and the checklist will guide the reviewers. A review summary will detail any changes that need to be made to the data design such as changes to data structures, types, keys or relationships.

**3.2.5.4 Review Data design checklist**

* Are all relationships between data structures properly made?
* Is the data model clear and comprehensive?
* Does the data design align with the requirements and use cases outlined in the SRS?
* Has input validation been incorporated with data types?
* Is the design scalable?

### 3.2.6 Architectural design review

**3.2.6.1 Description and focus of the review**

The review will focus on the architecture of the project or the blueprint created for the system. The review will ensure the architectural design clearly defines Moon Eyes structural elements and interfaces. Models will be reviewed at this stage as well.

**3.2.6.2 Timing of the review**

This will take place after the data design review and after the architectural design has been completed. This will take place in the design phase and should be revisited if architectural changes need to be made during the software development life cycle. Architecture can have a big impact on all other design elements.

**3.2.6.3 Work products produced**

The recorder will again take notes and develop a review summary report including decisions. It may also mention outcomes for design alternatives, modified elements, and more issues raised during the review. Additionally, an *Architecture Decision Description Template* can be found in the Pressman text that will be used as reference to make decisions that arose during the review. The checklist will also be referenced.

**3.2.6.4 Review Architectural design checklist**

* Does the architectural design address and align with all functional requirements?
* Does the architectural design address and align with all non-functional requirements like security/usability/etc.?
* Are the structures and elements in the architecture design clearly defined?

### 3.2.7 Interface (GUI) design review

**3.2.7.1 Description and focus of the review**

The GUI review will focus on the assessment of the visual clarity, cohesiveness, usability, accessibility, and aesthetics of the user interface. The review will ensure that the GUI aligns with the clients wants and is usable for the intended audience.

**3.2.7.2 Timing of the review**

This will take place during the design phase after the GUI has been designed. This will allow for early assessment of elements of the GUI design.

Additionally, this can be broken down into sub-reviews or smaller reviews for the various interfaces to be developed. (Like data submission pages, admin dashboard interface, etc.)

**3.2.7.3 Work products produced**

The recorder will take notes, reference the checklist given below and also create a summary report. The summary report will mention all concerns with the GUI and decisions on what actions should be taken (Like changing a certain color, font, etc.)

**3.2.7.4 Review Interface (GUI) design checklist**

* Are the design elements such as fonts, colors, and sizing consistent and legible?
* Does the GUI meet accessibility standards?
* Do interactive elements(such as the sidebar) react appropriately to user interaction?
* Is the GUI aesthetically pleasing?

### 3.2.8 Component design review(s)

**3.2.8.1 Description and focus of the review**

This review will focus on the design of each component for Moon Eyes. For more details on each component and what will be discussed for each please view the Moon Eyes Software Requirements Specification . We will evaluate the correctness of each component design to ensure that we have modular components before coding begins for that component.

**3.2.8.2 Timing of the review**

A FTR will take place during the design phase after the completion of the architecture design review. Smaller, less formal reviews can also take place for subcomponents in preparation for a major component design review. It needs to take place because it bridges the gap between architecture and coding.

**3.2.8.3 Work products produced**

The recorder will take notes for each review and also use the checklist as a guide. A review summary will also be developed which will touch on any concerns raised during the review

**3.2.8.4 Review Component designs checklist**

* Have all subcomponents and details listed in the architecture design been addressed in the component design?
* Does this component need to be broken into subcomponents?
* Are components well defined and separate so there is no overlap in development and is reusability noted?

### 3.2.9 Code Reviews

**3.2.9.1 Description and focus of the review**

The purpose of this review will be to ensure that any major code being pushed is accurate, understandable, quality, and in some cases reusable. This will also serve as a way for the group to review and make sure the code accomplishes the requirements.

**3.2.9.2 Timing of the review**

This will happen iteratively during development before code changes are pushed to the master branch in our version control system, Github.

**3.2.9.3 Work products produced**

Again, the recorder will take notes and address the questions on the below checklist. Additionally, the text also has a recommended code checklist from the given link: <https://courses.cs.washington.edu/courses/cse403/12wi/sections/12wi_code_review_checklist.pdf>

Also, the recorder will create a review summary that details any changes to be made such as code revisions, security maintenance, optimization and more.

**3.2.9.4 Review Code reviews checklist**

* Is the code readable and commented with clarity?
* Could the code be optimized better?
* Is there proper error handling in the code?
* Does the code have any security issues?

### 3.2.10 Test specification review

**3.2.10.1 Description and focus of the review**

The focus of this review will be the test specification plan. The main goal is to address the accuracy of the test cases and methods and ensure they align with the requirements and scope of Moon Eyes.

**3.2.10.2 Timing of the review**

This review will occur early in development after the test specification was written. The timing matters for early detection of any problems with test cases.

**3.2.10.3 Work products produced**

The recorder will take notes and utilize the checklist.

Additionally, a review summary will address any issues raised during the review and note any changes to be made such as a new or modified test case or a different testing method,

**3.2.10.4 Review Test specification checklist**

* Is each module tested fully?
* Is there multiple methods of testing such as block-box and white-box?
* Are any tests unnecessary?
* If a test is failed, are the mitigation strategies clear?

### 3.2.11 Change control reviews and audits

**3.2.11.1 Description and focus of the review**

This review will focus on the change control process and any large changes made to the project, maybe even a change discussed in a different FTR. This review will evaluate if the change is necessary and if it fixes the pre-existing issue.

**3.2.11.2 Timing of the review**

Change control reviews will be done whenever a change request is submitted and before any big changes are pushed to Github.

**3.2.11.3 Work products produced**

The producer will make an informal change request.

During the review the recorder will take notes and reference the checklist. A review summary report will be made with details on the decisions made and if any changes need to be made before the modification is pushed.

**3.2.11.4 Review Change control reviews and audits checklist**

* Were any alternative changes raised?
* Have risks been evaluated for the new change?
  + Does this change have the potential to impact another component?
* Does this change actually fix the pre-existing problems?

## 3.3 SQA Audits

The SQA audit will serve as a means to assess if our team is doing everything necessary to maintain high standards of software quality. We will discuss and assess how the FTR and informal reviews have been impacting our project. To do this we can have regular assessments(monthly) of our work products from each FTR and ensure that changes listed in the review summary reports have been acted on.

If we deem any part of the SQA process or formal technical reviews as: unhelpful, not useful, redundant or a time-waster, we will revisit that method going forward.

If any SQA process caught multiple errors and was useful to us we will continue that activity.

With a smaller 4-person team, the SQA audits will be less formal or complex compared to larger projects, but they are still necessary in order to maintain the quality of Moon Eyes and ensure that we are meeting our project requirements.

# 4.0 Problem Reporting and Corrective Action/Follow-up

This section describes problem reporting mechanisms that occur as a consequence of the FTR's that are conducted and the means for corrective action and follow-up.

## 4.1 Reporting mechanisms

Describes how and to whom problems are reported  
  
We will utilize the platform "Trello" for problem reporting. Trello is a free website ([www.trello.com](http://www.trello.com/)) that allows us to assign tasks to group members and track their progress. During the testing process, each member involved will log problems and bugs on Trello. The related members will receive tasks to address these issues, and once resolved, the corresponding card will be moved to the "Fixed Problems" list. The website is accessible from any location and on various devices, providing convenient access for all team members.

## 4.2 Responsibilities

Describes who has responsibility for corrective actions and follow-up.

1. **Software Quality Assurance (SQA) Team**

The SQA team plays a central role in identifying, documenting, and addressing defects and issues that arise during testing.After discussing with our team, we decided that Pamela Shahu and Kendall Gesek will cover this section.They are responsible for meticulously examining the software, executing test cases, and promptly reporting any deviations from the specified requirements or quality standards. The SQA team collaborates closely with the development team to communicate identified defects, providing detailed information that aids in the efficient resolution of issues. They also propose corrective actions and verify that the implemented solutions effectively address the reported problems. The SQA team monitors the progress of defect resolution, conducts retesting, and ensures that the software meets the required quality benchmarks.

1. **Development Team**

The development team takes ownership of rectifying the defects and issues reported by the SQA team. After discussing with our team, we decided that Kylie Callison and Erika Valle-Baird will cover this section. They prioritize the reported problems based on their severity and impact, dedicating resources to analyze and develop appropriate corrective solutions. The development team collaborates with the SQA team to gain a comprehensive understanding of the defects, including their underlying causes. They implement fixes, code changes, or patches to address the reported issues while ensuring that the software's overall functionality and stability are maintained. The development team also participates in root cause analysis to prevent similar defects from arising in the future.

1. **Stakeholders and End-Users**

Stakeholders, including end-users, have a responsibility to actively participate in the defect identification and resolution process. Due to the size of our team we have to play multiple roles and after discussing with our team, we decided that all four of us will cover this section.  
By providing timely and accurate feedback, we will contribute to the comprehensive evaluation of the software's performance. They also engage in user acceptance testing to validate the effectiveness of corrective actions and confirm that the reported issues have been successfully resolved.

1. **Documentation and Communication**

Documentation and communication are essential aspects of effective corrective actions and follow-up. The responsibility for documenting identified defects, proposed solutions, and the outcomes of corrective actions falls on both the SQA and development teams. Detailed records are maintained, including the steps taken to address each issue and the results of retesting. Clear and organized documentation ensures that all stakeholders have access to accurate information regarding defect resolution. Communication channels are established to provide updates on the progress of corrective actions, fostering collaboration and transparency among the teams involved.

1. **Continuous Monitoring and Review**

The responsibility for continuous monitoring and review of the corrective action process is shared among the SQA team, development team. This ongoing monitoring ensures that the implemented solutions are effective and that defects do not reoccur. If similar issues arise, the teams collaborate to identify root causes and adjust corrective strategies accordingly. Lessons learned from previous defect resolutions are incorporated into future development and testing processes, contributing to a cycle of continuous improvement.

## 

## 4.3 Data collection and evaluation

Describes the manner in which error/defect data are collected and stored for future or real-time evaluation.

Throughout the software development project, our team will actively involve clients and potential users to gather their feedback and preferences regarding system usability and user-friendly interfaces. We will conduct surveys to gather feedback from users, ensuring we take their input into account. Our testing team will diligently validate the implementation of requirements and user feedback to ensure smooth functionality. Any bugs discovered during the testing phase will be promptly reported to the development team and addressed accordingly.

* Data Collection and Evaluation

The systematic collection and evaluation of error and defect data are vital components of our quality assurance efforts for the web-based detective case management project. A well-defined process for gathering and storing such data enables us to proactively address issues, continuously improve our software, and ensure its overall reliability and user satisfaction. This section outlines the procedures and methodologies employed by the SQA team for the collection and storage of error/defect data, both for real-time assessment and future analysis.

* Defect Identification and Documentation

The SQA team is responsible for thoroughly documenting each reported defect using a standardized format. This documentation includes a clear description of the issue, steps to reproduce, observed behavior, and expected behavior. Additionally, defects are categorized based on severity, priority, and the specific module or functional area they pertain to.

* Defect Tracking System

To facilitate efficient management and tracking of defect data, a dedicated defect tracking system is utilized. This system acts as a centralized repository for all reported defects, streamlining communication and collaboration between the SQA team and the development team. The defect tracking tool allows us to log new defects, assign responsibilities, track progress, and maintain a complete audit trail of each defect's life cycle.

* Real-Time Monitoring and Alerts

For immediate response to critical issues, real-time monitoring tools are integrated into our software environment. These tools are designed to detect predefined thresholds or anomalies in the system's behavior. Upon detection, automated alerts are generated and sent to the SQA team, enabling timely intervention and investigation of urgent defects.

* Version Control Integration

Defect data is seamlessly integrated with our version control system, ensuring that each reported issue is associated with the specific software version or release in which it was identified. This integration allows us to analyze defect trends across different software iterations, track changes, and assess the effectiveness of defect resolution efforts over time.

* Root Cause Analysis and Documentation

Every reported defect undergoes a thorough root cause analysis conducted by the SQA and development teams. The analysis aims to identify the underlying reasons for the defect's occurrence. Results of the analysis, including identified root causes and potential corrective actions, are documented alongside the defect details. This documentation assists in preventing similar defects in the future and guides our continuous improvement efforts.

* Continuous Improvement Loop

The defect data collected and analyzed serve as a critical feedback loop for our continuous improvement efforts. Insights gained from defect analysis drive the identification of areas for enhancement in testing practices, development processes, and overall software quality. Regular reviews of defect data guide us in refining our strategies, methodologies, and best practices.

* Lessons Learned and Knowledge Sharing

The insights derived from defect data analysis are shared within the SQA team and across project teams. Lessons learned from defect resolution, root cause analysis, and process improvements contribute to a culture of knowledge sharing, fostering ongoing skill development, and promoting a proactive approach to defect prevention.

## 4.4 Statistical SQA

Describes the quantitative techniques that will be applied to error/defect data in an effort to discern trends and improvement.

To ensure the high-quality performance of the software we will use different data-based and statistical metrics in the testing phase. According to the mentioned metrics in the use case document, we will use the following metric to monitor and improve the software quality and performance to provide the best experience to the users of the system.

* User Adoption Rate
* User Engagement Metrics
* System Performance Metrics
* Error Rate
* Search Efficiency
* Security and Privacy Metrics

To monitor system performance, we will gather data in different formats and presentations. Additionally, any identified errors will be promptly reported to the team for priority fixing. Our goal is to continuously improve the software's performance and provide the best possible experience for system users.

# 5.0 Software Process Improvement Activities

The SQA group (and others) is often chartered with responsibility for software process improvement (SPI). This section describes the work associated with SPI.

## 5.1 Goal and objectives of SPI

The goal and objectives of SPI are defined.

SPI's (Software Process Improvement) Goal and Objectives are as follows:

* Implementing Suggested Changes: The primary goal is to implement all suggested changes in requirements to enhance software quality and reduce the error rate (number of bugs in the code).
* User Comfort and Interface Usability: Ensuring that all users find the system comfortable to use and that the interface is user-friendly, enabling them to accomplish desired tasks efficiently.
* Flexibility for New Functionality: Ensuring the system is capable of accommodating new functionality and features with ease, facilitating future enhancements.
* Timely Software Process: Ensuring that the software process and development cycle progress on schedule, delivering all the required components on time to avoid any mismanagement of time.
* Adherence to Allocated Budget: Ensuring that all process and development activities are within the allocated budget and do not exceed the agreed-upon price.
* Efficient Resource Management: Ensuring that resources are available as needed and procured within the budgetary constraints.
* Risk Identification and Mitigation: Identifying and effectively addressing all potential risks according to the Risk Mitigation, Monitoring, and Management (RMMM) Plan.
* Regular Monitoring and Performance Recording: Regularly monitoring all activities and system performance, recording the data to improve system performance and maintain the overall quality of the system.

By achieving these objectives, the Software Process Improvement (SPI) initiative aims to enhance the software's quality, user experience, and overall efficiency while adhering to the specified budget and schedule.

## 5.2 SPI tasks and responsibilities

Specific to the SQA group

Our commitment to excellence involves a structured approach to Software Process Improvement (SPI), aimed at enhancing our practices, methodologies, and overall effectiveness. The following tasks and responsibilities outline the SQA group's contributions to SPI efforts:

* **Process Evaluation and Assessment**

The SQA group is responsible for conducting thorough evaluations of our existing quality assurance processes. This includes regular assessments of testing methodologies, defect management, and adherence to established standards. The following tasks fall within this responsibility:

* Conduct Process Audits: Periodically assess our SQA processes to identify areas for improvement and ensure alignment with industry best practices.
* Analyze Process Performance: Review key performance indicators (KPIs) related to defect identification, resolution time, and test coverage to gauge process effectiveness.
* Identify Bottlenecks: Collaborate with the development team to identify bottlenecks and pain points in our quality assurance processes, and propose strategies for improvement.
* **Process Enhancement and Refinement**

To ensure continuous process improvement, the SQA group actively engages in refining existing methodologies and adopting industry-leading practices. The following responsibilities are integral to this effort:

* Recommend Best Practices: Research and recommend industry best practices, testing frameworks, and tools to enhance our testing processes.
* Develop Guidelines and Standards: Collaborate with the SQA team to establish standardized guidelines and practices for test case design, defect reporting, and documentation.
* Implement Process Changes: Lead the implementation of approved process changes, ensuring seamless integration into our workflow and effective adoption by the team.
* **Data-Driven Decision-Making**

Leveraging data to drive decisions is fundamental to our SPI efforts. The SQA group plays a crucial role in analyzing defects and testing data to inform improvements. The following responsibilities contribute to data-driven decision-making:

* Analyze Defect Trends: Regularly analyze defect data to identify trends, patterns, and areas of concern that warrant further investigation or process enhancement.
* Generate Performance Reports: Produce reports showcasing defect trends, test coverage, and testing efficiency metrics to facilitate informed decision-making.
* Provide Insights: Collaborate with the project manager and development team to provide insights from defect analysis that guide process improvements and resource allocation.
* **Training and Knowledge Sharing**

Elevating the skill set of the SQA team and fostering a culture of knowledge sharing are essential components of SPI. The following responsibilities focus on continuous learning and skill development:

* Develop Training Programs: Create training programs that enhance the testing skills and expertise of SQA team members, covering testing methodologies, tools, and best practices.
* Facilitate Workshops: Organize workshops and knowledge-sharing sessions to promote the exchange of insights, lessons learned, and emerging industry trends.
* Encourage Cross-Functional Collaboration: Foster collaboration between SQA team members and other project teams to share knowledge and benefit from diverse expertise.
* **Process Monitoring and Adaptation**

SPI involves ongoing monitoring and adaptation of our quality assurance processes. The SQA group is responsible for ensuring that process improvements are effective and aligned with project goals:

* Evaluate Process Impact: Continuously assess the impact of implemented process improvements on defect identification, resolution, and overall software quality.
* Adjust Strategies: Based on data and feedback, refine and adjust SPI strategies to ensure ongoing alignment with project objectives and evolving industry standards.
* Facilitate Process Reviews: Organize regular reviews and retrospectives to evaluate the effectiveness of SPI efforts and identify areas for further enhancement.
* **Stakeholder Collaboration**

SPI efforts extend to collaboration with stakeholders to ensure that the project's quality goals are met. The SQA group plays a pivotal role in this collaboration:

* Collaborate with Stakeholders: Engage with stakeholders, project managers, and developers to align SPI initiatives with project priorities, objectives, and expectations.
* Share Improvement Progress: Provide transparent updates on SPI progress, sharing successes, challenges, and the impact of process enhancements.
* Gather Feedback: Solicit feedback from stakeholders to identify their perspectives on the effectiveness of process improvements and areas for additional focus.

In conclusion, the SQA group's SPI tasks and responsibilities are integral to our commitment to continuous improvement and software quality excellence. By conducting process evaluations, enhancing methodologies, leveraging data, promoting learning, adapting strategies, and collaborating with stakeholders, the SQA group contributes significantly to the success of our SPI initiatives and the overall project's quality outcomes.

# 6.0 Software Configuration Management Overview

Software Configuration Management (SCM) 's focal point is identifying and controlling major software changes as well as recording and reporting the status of the changes. In order to achieve uniformity and proper implementation, the team will use a systematic approach to manage, facilitate, and authorize changes to the software system throughout its life cycle. This life cycle is described in the Software Requirements Plan with the requirements, implementation, test, and acceptance stated This will allow us to ensure the system is compliant with all functional requirements and meets all previously set standards while limiting the impact any changes can have on the entire system. When a change is identified in the software system pertaining to one of the functional requirements, the change request identification process begins methodically to correct the defect in the system. The development team will be responsible for managing all changes to the system. When this occurs, all changes are evaluated by the entire team because of the team’s size and documented for future reference to prevent the issue from recurring.

At the beginning of the change request identification, both issue resolution and an impact summary are required to determine the steps that need to be taken to resolve the issue with a focus on how this will impact our system in the future. At this point, a change request is logged into the system to track and ensure proper documentation. From there, the team reviews and evaluates the change request to decide how the team will move forward with the change and the viability of it. If the change is deemed beneficial to the program, implementation will occur as well as an update to the log. If the change is denied, a revision can be made to the proposed change request to be reassessed by the team at a later time. If the change is approved, at this point in the process any resources needed for the change will be discussed by the team and added to documentation to ensure completion. The change will be evaluated in a testing area before being implemented into the software system to ensure the product is performing as intended. Once the change has been implemented into the system, the maintenance of the change will fall on to the team and verifying all documentation reflects the completion of the change.

# 7.0 SQA Tools, Techniques, Methods

**Software Quality Metrics**

*Customer Satisfaction*

One of the product quality metrics we plan to employ is customer satisfaction. This metric will be measured using a five-point scale: Very Satisfied, Satisfied, Neutral, Dissatisfied, Very Dissatisfied. To obtain this metric, surveys will be carried out for each aspect of software that is developed.

*Fix Backlog and Backlog Management Index*

This maintenance quality metric will be employed using a Trello board, allowing our team to fix defects based on their priority in the backlog. To increase customer satisfaction, customers have access to the aforementioned board and are able to inform our team about requested fixes at any time, leading our team to maintain shorter fix response times.

*Fix Quality*

Frequent client meetings allow our team to be informed about defects as soon as they are encountered. By eliminating defects as soon as possible, we are able to fix quality, response time, and responsiveness. This quality maintenance metric will be determined by the rate of defective fixes without delinquency. As there are many different schedules that conflict with all member participation, meeting notes will be taken and distributed to all members after each meeting.

*Quality Indicators*

To measure and ensure software quality, the SQA team plans to employ the following quality metrics:

Reliability

Portability

Efficiency

Maintainability

Usability

Security

Performance

*Methods*

To better align ourselves with our customer’s needs, we will be enlisting the ISO 9000’s seven principles of quality management.

***These principles include:***

Customer Focus

Leadership

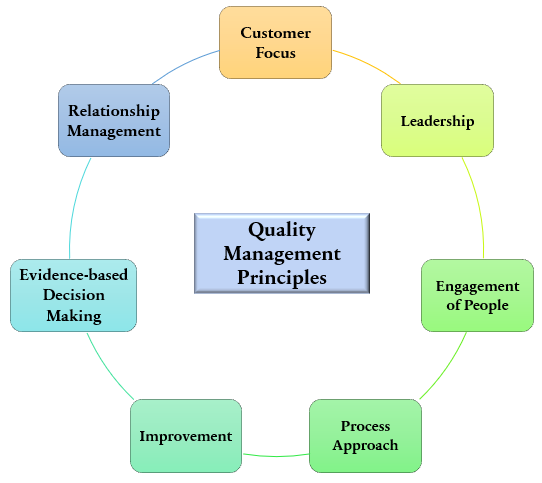
Engagement of people

Process approach

Improvement

Evidence-based decision making

Relationship management

****

*Customer Focus*

With an emphasis on customer focus, our team will be better able to meet customer requirements, measure our customer’s satisfaction with our software quality metrics, and be better adapted at understanding our customer’s needs.

*Leadership*

By setting and reaching our goals, we can establish trust both within our software team, and with our customers.

*Engagement of People*

Shared drives, group messages, and meetings ensure learning and information sharing, as well as allowing our team to enlist the right person for each task.

*Process Approach*

By enlisting a process approach, we can establish activities as processes, allowing our team to prioritize improvements, and manage activities.

*Improvement*

We will provide Improvement by measuring and empowering our team members to improve our software quality.

*Evidence-Based Decision Making*

Decisions will be made based on analysis, appropriate methods, and employed tools.

*Relationship Management*

By constant communication with our clients, our relationship management will consist of honest and open dialogue.

*In addition to this, techniques we plan to employee consist of:*

Auditing

Reviewing

Code Inspection

Design Inspection

Simulation through prototype

Functional Testing

Standardization (IEEE)

Walkthroughs

Unit testing

Stress testing

*Auditing*

Auditing will be instituted via inspecting work submitted against a set of standard processes to determine if they were followed.

*Reviewing*

Meetings are set up to review all software processes.

*Code Inspection*

Static testing will be established to find bugs and avoid defects.

*Design Inspection*

A checklist will be employed to ensure design inspection can be done efficiently. Such items will include general requirements, logic, structures, interfaces, and requirement traceability.

*Simulation*

A simulation will be completed through the creation of a prototype.

*Functional Testing*

Black box testing will be conducted to test the system specifications and features.

*Standardization (IEEE)*

Standardization will be required for all documentation and work completion to institute consistent standards and practices.

*Walkthroughs*

Software walkthroughs will be arranged to guide all team members in the development process.

*Unit testing*

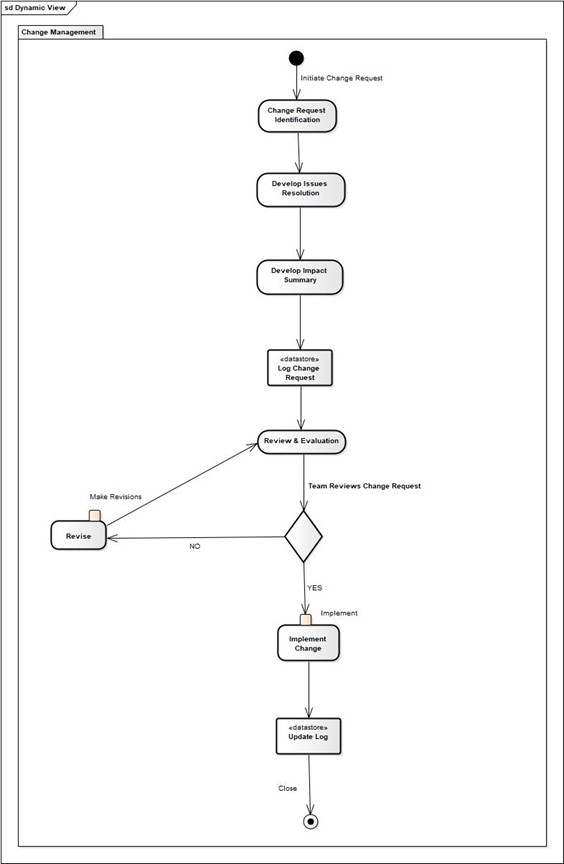
To secure complete code coverage, White Box Testing will be executed on each path, branch, and condition.

*Stress testing*

To ensure our product works under normal conditions, stress testing will be completed to test the robustness of our system.

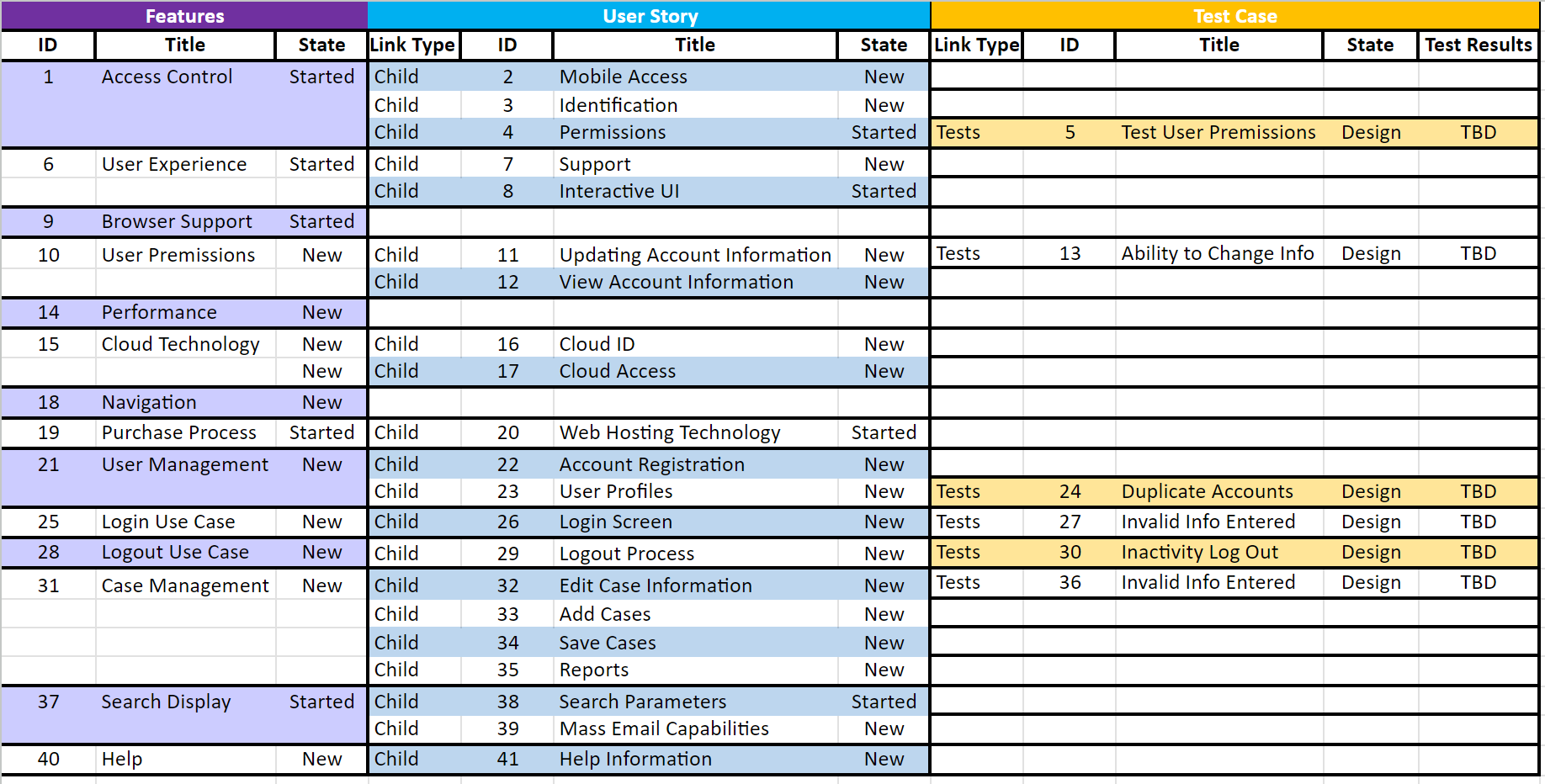
*Change Management Method*

In addition to this, provided below is a visual diagram of the method we will employ for all change management activities. Further information on this can be found in the Software Requirements Specification.



# 8.0 Appendix

## 8.1 Traceability Matrix



## 8.2 References

This project plan was created using the latest Pressman format. Additionally, the Cyber Rovers “WMITS Software Quality Assurance Plan” and an additional “Software Quality Assurance” found on McGraw Hill website were used as reference and are cited below.

The textbook *Software Engineering: A Practitioner's Approach* was also referenced throughout the creation of this document.

* R. S. Pressman, “Document Templates: Software Quality Assurance Plan,” *R.S. Pressman & Associates, Inc..* [Online]. Available: http://www.rspa.com/docs/sqaplan.html. [Accessed: 14-July-2023].
* Pressman, Roger S., and Bruce R. Maxim. *Software Engineering: A Practitioner's Approach* . Vol. 9, McGraw-Hill Education , 2020.
* R. S. Pressman, “Document Templates: SQA Plan,” *R.S. Pressman & Associates, Inc.*. [Online]. Available: http://www.rspa.com/docs/sqaplan.html.[Accessed: 14-July-2023].
* “Software Quality Assurance Plan,” *McGraw Hill Higher Education*. [Online]. Available: http://www.mhhe.com/engcs/compsci/pressman/graphics/Pressman5sepa/common/cs1/sqa.pdf.[Accessed: 14-July-2023].
* “Software Quality Assurance Plan,” *McGraw Hill Higher Education*. [Online]. Available: http://www.mhhe.com/engcs/compsci/pressman/graphics/Pressman5sepa/common/cs2/sqa.pdf. [Accessed: 14-July-2023].