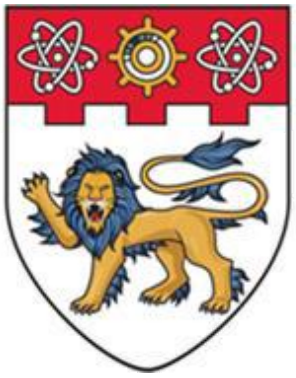

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Software Quality Assurance (SQA) Plan for QMePls

By Team Titans

Date: 9 September 2021

Seow Jing Hng Aloysius
Jacob Law Zhen
Jolene Tan
Soh Qian Yi
Samuel Lee Si En
Zeta Chua Hui Shi

Team Titans
School of Computer Science and Engineering

Signature Page

Prepared by: Aloysius Seow **Date: 09/09/2021**

Reviewed by 1: Jacob Law **Date: 21/09/2021**

Reviewed by 2: Samuel Lee **Date: 21/09/2021**

Reviewed by 3: Soh Qian Yi **Date: 21/09/2021**

Reviewed by 4: Zeta Chua **Date: 21/09/2021**

Approved by : Jolene Tan **Date: 22/09/2021**

Document Change Record

Revision	Revised By	Description of Change	Approved by	Date
0.10	Jolene	Initial Template	Aloysius	<09/09/21>
0.20	Aloysius	Purpose and Scope	Jolene	<10/09/21>
0.30	Jacob	Management	Aloysius	<11/09/21>
0.40	Qian Yi	Documents	Aloysius	<12/09/21>
0.50	Samuel	Standards, Practices, Conventions and Metrics	Aloysius	<13/09/21>
0.60	Zeta	Software Reviews	Aloysius	<14/09/21>
0.70	Jolene	Test	Aloysius	<15/09/21>
0.80	Aloysius	Problem Reporting and Corrective Action	Jolene	<16/09/21>
0.90	Jacob	Tools, Techniques and Methodologies Media Control	Aloysius	<17/09/21>
1.00	Qian Yi	Record Collection, Maintenance, and Retention	Aloysius	<18/09/21>
1.10	Samuel	Training Risk Management	Aloysius	<19/09/21>
1.20	Zeta	SQA Plan Change Procedure and History	Aloysius	<20/09/21>

Contents

1. Purpose and Scope

1.1. Purpose

The purpose of this Software Quality Assurance (SQA) Plan is to establish the goals, processes, and responsibilities required to implement effective quality assurance functions for the QMePls project.

The Software Quality Assurance Plan provides the framework necessary to ensure a consistent approach to software quality assurance throughout the project life cycle. It defines the approach that will be used by the QAM and Software Quality (SQ) personnel to monitor and assess software development processes and products to provide objective insight into the maturity and quality of the software. The systematic monitoring of products, processes, and services will be evaluated to ensure they meet requirements and comply with policies, standards, and procedures, as well as applicable Institute of Electrical and Electronic Engineers (IEEE) and ISO standards.

1.2. Scope

The purpose of SQA is to ensure that the software developed does not deviate from the original intended product. SQA is also concerned to identify any errors, omissions, inconsistencies, and alternatives, enhancements or improvements that can be made at any stage of development.

QMePls is an application meant for patients who possess a personal mobile device to provide convenience when planning to visit a clinic. This is done by providing the app users relevant information on the clinics such as number of patients waiting in queue, so that users can better manage their time instead of having to waste a trip down to a clinic and having to wait long hours.

2. Reference Documents

- IEEE STD 730-2002, IEEE Standard for Software Quality Assurance Plans (http://standards.ieee.org/reading/ieee/std_public/description/se/730-2002_desc.html)
- ISO IEC 90003:2004 Software Standard (http://praxiom.com/iso-90003.htm)
- Project Plan
- System Requirement Specifications

3. Management

This section describes the management organizational structure, its roles and responsibilities, and the software quality tasks to be performed.

3.1. Management Organisation

The implementation of quality assurance system is the responsibility of the Quality Assurance Manager (QAM).

3.1.1. Project Management

The Project Manager will be responsible for approving:-

- The system requirement specification document
- The overall time scale for the project
- The choice of system development life cycle
- The choice of software development tools and techniques utilised
- The selection of project teams
- The training of project teams

3.1.2. Assurance Management

The QAM provides Project Management with visibility into the processes being used by the software development teams and the quality of the products being built. The QAM maintains a level of independence from the project and the software developers.

In support of software quality assurance activities, the QAM has assigned and secured Software Quality personnel from the pool of available SQ trainees to coordinate and conduct the SQ activities for the project and report back results and issues.

3.2. Tasks

This section summarizes the tasks (product and process assessments) to be performed during the development of software. These tasks are selected based on the developer's Project Plan and planned deliverables, and identified reviews.

3.2.1. Product Assessments

The following product assessments will be conducted by SQ personnel:

- Weekly checklist for functional requirements
- Weekly checklist for non-functional requirements

3.2.2. Process Assessments

The following process assessments will be conducted by SQ personnel:

- Weekly code review to be in line with software metrics
- Weekly progress check for QMePls against proposed Gantt Chart timeline
- Weekly meeting to verify adherence to requirements specification
- Comply with change management plans when change is required
- Daily task update by project members for task completion

3.3. Roles and Responsibilities

This section describes the roles and responsibilities for each assurance person assigned to the Project.

3.3.1. QAM

Responsibilities include, but are not limited to:

- Secure and manage SQ personnel resource levels
- Ensure that SQ personnel have office space and the appropriate tools to conduct SQ activities
- Provide general guidance and direction to the SQ personnel responsible for conducting software quality activities and assessments
- Assist SQ personnel in the resolution of any issues/concerns and/or risks identified as a result of software quality activities
- Escalate any issues/concerns/risks to project management

3.3.2. Software Quality Personnel

Responsibilities include, but are not limited to:

- Develop and maintain the project software quality assurance plan
- Generate and maintain a schedule of software quality assurance activities
- Conduct process and product assessments, as described within this plan
- Identify/report findings, observations, and risks from all software assurance related activities to the QAM

4. Documents

4.1. Purpose

This section identifies the minimum documentation governing the requirements, development, verification, validation, and maintenance of software that falls within the scope of this software quality plan. Each document below shall be assessed (reviewed) by SQ personnel.

4.2. Minimum Document Requirements

- System Requirement Specifications: Based on the use case models, System Requirement Specifications specifies the main purpose and direction of the system. It also lists the critical constraints, high-level functional requirements and non-functional requirements of the system and major features of the system in an atomic, unambiguous, traceable and verifiable manner.
- Quality Assurance Plan: The Quality Assurance Plan contains a set of documented activities meant to ensure our customer is satisfied with the goods and services provided by QMePls. This document defines the objectives, role and responsibilities, coordinates with other plans and defines tasks and schedules.
- Risk Management Plan: The Risk Management Plan identifies potential risks to the project (risk identification), estimates the impact and the probability of them happening (risk analysis) and then defines response.
- Configuration Management Plan: The configuration management plan defines those items that are configurable, those items that require formal change control and the process for controlling changes to such items.
- Change Management Plan: The Change Management Plan is used when changes are made to the existing products and processes to ensure effective implementation of the changes, which helps

manage the change process and ensures control in budget, schedule, scope, communication and resources.

- Release Management Plan: The Release Management Plan defines software deployments and change initiatives. This includes scheduling the relevant tasks (internal and external), assigning the physical and human resources needed to carry them out, and overseeing the execution.

5. Standards, Practices, Conventions and Metrics

5.1. Purpose

This section highlights the standards, practices, quality requirements, and metrics to be applied to ensure a successful software quality program.

5.2. Software Quality Programme

These practices and conventions are tools used to ensure a consistent approach to software quality for all programs/projects.

Qualities	Description	Considerations
Availability	The degree to which a system is in a specified operable and committable state. Normal high availability ratings are specified at 99.999% uptime per year.	Instance of QMePls in its production phase will be deployed in Firebase & Firestore instances, thus guaranteeing availability on Google's highly available and fault tolerant data centers with emergency fall back systems in place.
Compatibility	The ability for a system to execute on different types of computers without modification of the program or the computers.	Given that QMePls is a phone application, as long as the client has an android device, they will have a compatible system capable of using 100% of QMePls's feature set regardless of device model and screen size.
Debuggability	The quality of a software to be debuggable with relative ease.	QMePls’s development cycle practices the Test Driven Development (TDD) principle. Development and testing occurs concurrently.

		Functions are written with their respective unit test functions, thus ensuring that testing can take place instantly hence boosting debuggability.
Fault-Tolerance	The property that describes whether a system continues operating properly in the event of the failure of some of its components.	QMePls operates with Firebase & FireStore both of which are maintained by Google, therefore ensuring fault tolerance with Google's reliable fall back mechanisms should their data centers fail.
Interoperability	If two or more systems use a common data format and communication protocol and are capable of communicating with each other, they exhib	The QMePls development team has established that data transmitted is standardized to be in the JSON format while Firebase requests and responses from are sent and received via the HTTP(S) protocol.
Securability	The degree to which a system is securable, especially of the ability of a system to provide different levels of secure access.	QMePls utilizes the Firebase Authentication that leverages industry standards like OAuth 2.0 and OpenID Connect. User-based permissions are granted, allowing the development team to set highly complex rules for the application.

5.2.1. Standard Metrics

The following standard metrics are the minimum planned metrics that will be collected, reported, and maintained in the area of software quality assurance:

Software Metric	Description	Considerations
Fan-in/Fan-out	Fan-in is a measurement of the number of functions or methods that call another function, or method. Fan-out is the number of functions that are called by other functions.	Java will be used for the frontend which follows object-oriented concepts and hence function calls made will be high. Ensuring the function is only called only when necessary is a key priority.

Length of Code	This is a measure of the size of a program. The larger the size of the code of a component, the more complex and error-prone the component is likely to be.	This is a size-oriented metric and is independent of the programming languages. Our frontend and backend codes should be considered separately.
Cyclomatic Complexity	This measures the control complexity of our project. With lower complexity, our program would be more understandable and readable.	Reducing code complexity improves code readability. As such, reducing cyclomatic complexity is a key priority.
Length of Identifiers	This measures the average length of distinct identifiers in the program. The longer the identifiers, the more meaningful and understandable the program is.	With longer and readable identifiers, it would provide more meaning for the users to associate and to understand.
Depth of Conditional Nesting	This is a measure of depth of nesting of if-statements in a program. Deeply nested if statements are hard to understand and are potentially error-prone.	Identifying deeply nested statements and proposing different methods would simplify the code and reduce potential errors.
Fog Index	This is a measure of the average length of words and sentences in documents. The higher the value for the Fog Index, the more difficult the document is to understand.	Reducing the average length of words and sentences used in our documentations are crucial. Hence, keeping the Fog Index low ensures the readability and understandability of our documents.

6. Software Reviews

6.1. Purpose

This section identifies the number and type of system/subsystem reviews and engineering peer reviews that will be supported by the SQ Personnel. The project milestone chart, and the SQ Personnel resource levels determine the reviews that are supported.

6.2. Minimum Software Reviews

For each review, SQ will assess the review products to assure that review packages are being developed according to the specified criteria, the review content is complete, accurate, and of sufficient detail, and Requests for Action are captured, reviewed, and tracked to closure. In addition, SQ will assess the processes used to conduct the reviews to determine if appropriate personnel are in attendance, correct information is presented, entry and exit criteria are met, and appropriate documents are identified for update.

The following software reviews will be assessed by SQ:

- Project Plan Review
- Requirements Analysis Review
- Software Design Review
- Test Plan Review
- Acceptance Review

7. Test

SQ personnel will assure that the test management processes and products are being implemented per Test Plan. This includes all types of testing of software system components as described in the test plan, specifically during integration testing (verification) and acceptance testing (validation). SQ personnel will monitor testing efforts to assure that test schedules are adhered to and maintained to reflect an accurate progression of the testing activities. SQ will assure that tests are conducted using approved test procedures and appropriate test tools, and that test anomalies are identified, documented, addressed, and tracked to closure. In addition, SQ will assure that assumptions, constraints, and test results are accurately recorded to substantiate the requirements verification/validation status. SQ personnel will review post-test execution related artifacts including test reports, test results, problem reports, updated requirements verification matrices, etc.

8. Problem Reporting and Corrective Action

SQ personnel generate, track, and trend assessment findings and observations in a centralized Reporting and Corrective Action System.

When a problem or failure is identified, the Reporting and Corrective Action plan that has been created in accordance with the Corrective Action Procedure (CAP) will be followed to eliminate any future occurrences of the same or similar failure.

The Reporting and Corrective Action plan has to be approved by both the QA Manager and the Project Manager before execution. The QA personnel will then be responsible for carrying out the plan in accordance with the specified problem/failure reported. The steps of the Reporting and Corrective Action plan are outlined below.

1. Review Failure History to look for related failure cases

2. Check if any existing solution can be used

3. Reenact & Document the cause of failure

4. Identify & Document all possible solutions

5. Document the cause of failure and actions taken for future reference

6. Test & Ensure that the problem is solved by recreating the cause of failure and using the identified solution

7. Request approval from the QA Manager and Project Manager to modify the existing system

8. Document the change log with regards to this current failure case and the identified solution

9. Resolve issue

9. Tools, Techniques and Methodologies

SQ personnel will require access to the following:

9.1. Software Quality Tools

- Android Studios
- Github
- Trello
- Firebase
- Figma

10. Media Control

SQ deliverables will be documented in one of the following Microsoft software applications: Word, Excel, or PowerPoint. Deliverables will be in soft copy, with the exception of completed checklists from process and product assessments. See Section 12 for additional details on the collection and retention of key records. Software Quality personnel will request space on the project's secured server for SQ records. This server is password protected and backed up nightly.

SQ Personnel will be utilizing Github as their main collaborative platform for code maintenance and back up.

11. Record Collection, Maintenance, and Retention

SQ personnel will maintain records that document assessments performed on the project. Maintaining these records will provide objective evidence and traceability of assessments performed throughout the project's life cycle. There are two types of records that will be maintained: Hardcopy and Electronic. SQ personnel will maintain electronic or hard copies of all assessment reports and findings. SQ Project folders will contain hardcopies of the assessment work products such as completed checklists, supporting objective evidence, and notes.

The table below identifies the record types that will be collected, as well as the Record Custodian and Retention period

Record Title	Record Custodian	Record Retention
SQA Assessments	SQ Personnel	One Year
SQA Checklists	SQ Personnel	One Year
Deliverable Defects	SQ Personnel	One Year

12. Training

SQ personnel have fundamental knowledge in the following areas through prior experience, training, or certification in methodologies, processes, and standards:

- Audits and Reviews (Assessments)
- Risk Management
- Software Assurance
- Configuration Management
- Software Engineering
- ISO 9001, ISO 9000-3
- CMMI
- Verification and Validation

13. Risk Management

SQ personnel will assess the project's risk management process and engage in quarterly risk management meetings, progress and report any software risks to the QAM and the project manager.

14. SQA Plan Change Procedure and History

SQ personnel are responsible for the maintenance of this plan. It is expected that this plan will be updated throughout the life cycle to reflect any changes in support levels and SQ activities. Proposed changes shall be submitted to the Quality Assurance Manager (QAM), along with supportive material justifying the proposed change.