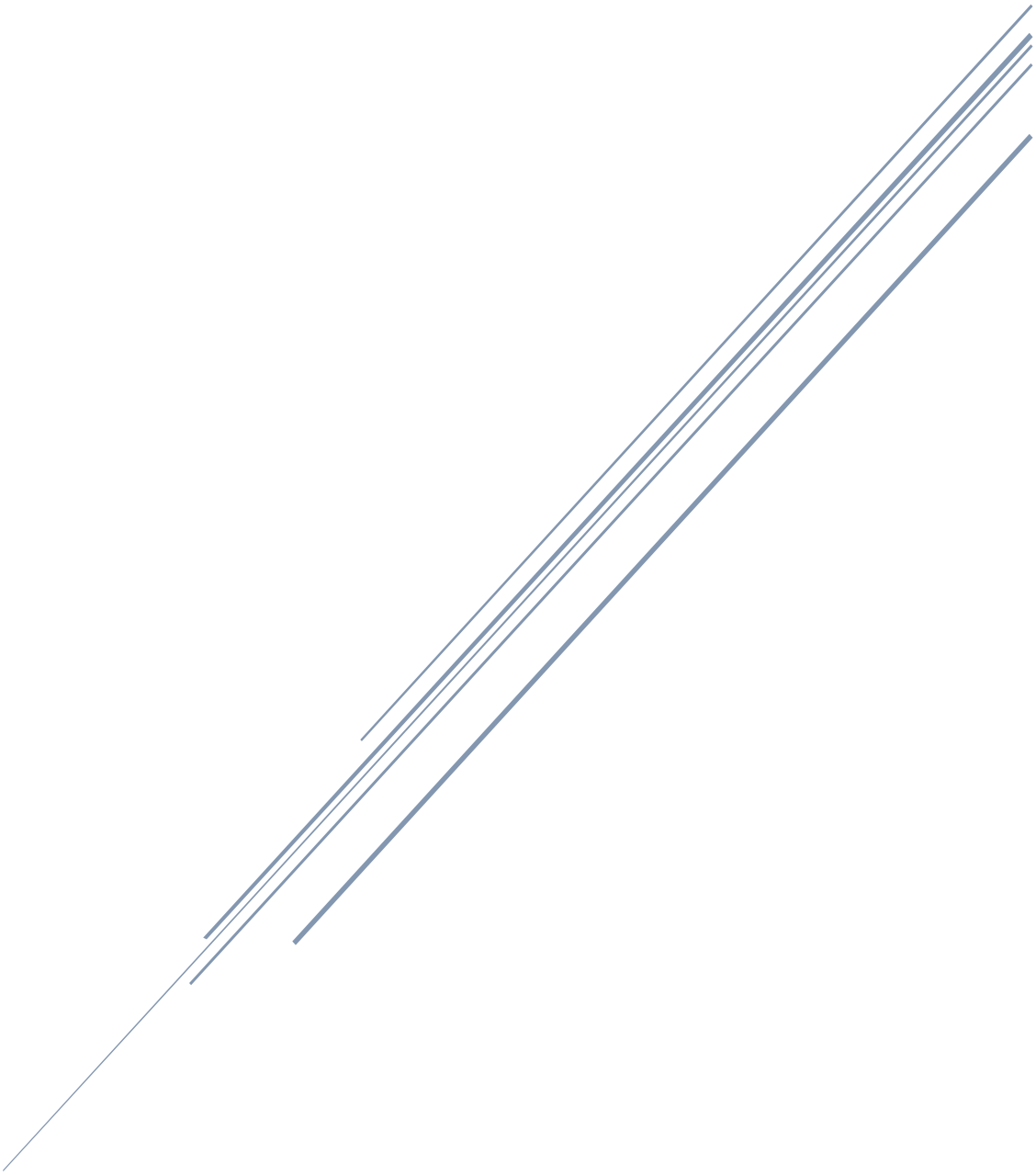
PROJECT PROPOSAL: GOTOGRO-MRM

Definition of Done and Quality



SWE20001 – Managing Software Projects

Table of Contents

1 Quality Management:

The “Definition of Done” for this project outlines the conditions that need to be met to ensure that our proposed solution fulfils both the functional and non-functional (quality) requirements as specified in the brief. The following 10 conditions, in alignment with the product quality model (Standards Australia/New Zealand, 2013), provide a more detailed definition of the "Definition of Done".

1.1 Functional Suitability:

1.1.1 Functional Correctness:

1.1.1.1 Number of Defects per KLOC (Thousand Lines of Code):

This metric quantifies the number of defects or bugs found in the codebase of the software per thousand lines of code. The threshold value is set to a maximum of **10 defects per KLOC**. This threshold value considers the trade-off between achieving a reasonable level of quality while acknowledging potential time and resource limitations (such as developers’ experience).

Example: After reviewing the codebase, it was determined that there were 5 defects found in 800 lines of code. This translates to 6.25 defects per KLOC, which falls within the acceptable threshold of 1-10 defects per KLOC.

1.1.2 Functional Completeness:

1.1.2.1 Coverage of Specified Tasks and User Objectives:

This metric evaluates the extent to which the set of functions implemented in the software addresses all the tasks and objectives specified in the project requirements. The threshold value is set to a **minimum of 80%**. Choosing the threshold of 80% provides flexibility and recognizes that software projects frequently involve iterations and potential improvements. This well-balanced approach ensures that the software accomplishes a majority of the intended tasks and objectives, while also permitting refinements and enhancements in subsequent iterations.

Example: The requirements include tasks like registering members, recording sales, managing products, and generating reports. During testing, it was found that the software effectively handles 4 out of 5 tasks, with minor UI improvements needed for product management. The software then meets the above threshold of 80% functional completeness.

1.2 Performance Efficiency:

1.2.1 Time Behaviour:

1.2.1.1 Response and Processing Times:

This metric assesses the speed at which the system responds to user actions and processes tasks. The threshold value is set at **2 seconds**. If the system responds to user actions within this time frame, it meets the quality metric.

Example: One of the key functionalities in GotoGro is adding a new member to the database. This involves capturing member information, validating it, and storing it in the database. The response and processing times for this action are critical to ensure a smooth user experience and efficient member management. After a user has entered the new member's information, the system verifies the input, checks for duplicate entries, and stores the new member's information in the database. If the system responds to the staff member's action and completes the entire process of adding a new member within 2 seconds, then this metric is met.

1.2.2 Resource Utilization:

This metric monitors the utilization of system resources such as CPU and memory. The threshold value is set at **70%**. If the system's resource utilization remains below 70% under normal load conditions, it meets this quality metric.

Example: when generating detailed inventory reports and sales analyses, the system's resource utilization needs to be monitored closely to guarantee that report generation is swift and doesn't hinder other critical system functionalities.

1.3 Usability

1.3.1 User Protection Error:

1.3.1.1 User Task Success Rate:

This metric evaluates the percentage of user tasks that are completed successfully without errors or assistance. The threshold value is set at **95% or higher**. This entails providing clear instructions and easy-to-fill forms for registration. If at least 95% of user tasks can be completed successfully by users without errors or difficulties, the software meets this quality metric.

Example: Out of 100 attempts to update member information, if at least 95 of them are completed successfully without any errors or the need for assistance, the software meets the quality metric.

1.3.1.2 Time to Complete Common Tasks:

This metric measures the time taken by users to complete common tasks within the user interface. The threshold value is set at **1.5 times the average time taken by experienced users**. If users can complete common tasks within this threshold, the software meets this quality metric.

Example: Suppose the average time taken by experienced users to create a new sale record is 2 minutes. In this case, the threshold value would be 1.5 times this average, which is 3 minutes. If most users can complete the sale record creation process within 3 minutes, the software meets the quality metric.

1.4 Reliability:

1.4.1 Availability:

1.4.1.1 System Uptime and Downtime:

This metric tracks the amount of time the system is operational (uptime) and the amount of time it is unavailable (downtime). The threshold value is set to **95% uptime**. The system should be operational and accessible for at least 95% of the time.

1.4.1.2 Mean Time to Recovery (MTTR):

This metric evaluates the average time it takes to restore the system to full functionality after a failure. The threshold value is set at **1 hour**. If the system can be fully restored within 1 hour after a failure, it meets this quality metric.

Example: If a technical glitch makes GotoGro's interface unresponsive, staff and members can't access the system, disrupting member management and grocery services. To meet the MTTR metric, GotoGro's tech team must quickly diagnose and fix the issue, restoring full system functionality within an hour, to achieve this quality metric.

1.5 Security:

1.5.1 Integrity:

1.5.1.1 Vulnerabilities Detected:

This metric identifies critical vulnerabilities found in the software through security assessments. The threshold value is set at **zero** (keeping in mind the size of this system, there shouldn’t be any critical vulnerabilities). If no critical vulnerabilities are detected in the software, it meets this quality metric.

Example: For GotoGro's project, a security assessment is conducted to identify vulnerabilities in the application. During the assessment, the team uses a reputable security scanning tool to analyse the application's codebase and dependencies. The assessment reveals one vulnerability with high severity. This vulnerability is promptly addressed and patched by the development team. Upon second

assessment, if it's found that there are no vulnerabilities, then the system passes this quality metric, otherwise, the development team needs to do more patching.

1.5.2 Confidentiality:

1.5.2.1 Data Encryption:

This metric ensures that sensitive data is encrypted both when at rest and during transmission. A reasonable threshold value is met if **sensitive data is encrypted during transmission** (e.g., using HTTPS) and stored using industry-standard encryption practices.

Example: To ensure the security of sensitive member data, GotoGro's application employs encryption practices. When members submit their information or make transactions, the data is encrypted during transmission using HTTPS, providing a secure connection between the user's browser and the server. Additionally, sensitive data, such as member details and sales records, is stored in the database using industry-standard encryption techniques to prevent unauthorized access. This approach ensures that sensitive information remains confidential and secure throughout its lifecycle.

2 References:

Standards Australia/New Zealand, 2013, *Systems and software engineering - Systems and software quality requirements and evaluation (SQuaRE) - System and software quality models,* AS/NZS ISO/IEC 25010:2013, Sydney/Wellington: Standards Australia/New Zealand, viewed 18 August 2023.