**Software Quality Assurance Plan (SQAP)**

**Revision History**

Version 1

1.0.0 Josh Potrawski & Justin Andras: created 2/25/17

**Table of Contents**

1. Purpose
2. Reference Documents
3. Management
4. Documentation
5. Standards
6. Reviews and Audits
7. Test
8. Problem Reporting and Corrective Action
9. Tools, Technique, and Methodology
10. Code Control
11. Media Control
12. Supplier Control
13. Records Collection, Maintenance, and Retention
14. Training
15. Risk Management

**1. Purpose**

This document plan (SQAP) will serve to provide the ways in which our University Website project will produce and maintain a high-quality product. It will also describe the methodologies, techniques, measurements, and mechanisms that we may potentially use to improve the quality of our product and the quality assurance process itself. The scope of this SQAP will cover most, if not all, aspects of the quality assurance process.

Using this document, we will determine if our product meets our set standards of quality and will work accordingly to resolve any issues.

**2. Reference Documents**

See Section 4.2.

**3. Management**

This section shall describe organization, tasks, and responsibilities.

3.1 Organization

The quality assurance leaders will consist of the two Testing Engineers in the team, since their roles are directly related to quality assurance. They will set the appropriate standards for the quality levels this team hopes to achieve. However, all decisions and ideas must be approved (with feedback provided) by the Project Manager.

The other team members will address their specific quality assurance questions to the QA leaders (made up of the testing engineers), who in turn will deliver potential solutions, provided that they have been approved by the project manager.

3.2 Tasks

General QA tasks include:

* Maintaining this document
* Documenting the quality of product as it evolves
* Developing advanced standards and metrics to measure product’s quality
* Determining defects/KLOC and their severity
* Sharing known defects at team meetings
* Discussing solutions for defects
* Developers responsible for ensuring their work meets team quality standards
* Project Manager declares final QA standards
* Inspections, unit tests, feedback, reviews

3.3 Responsibilities

It is the responsibility of every Code Developer to ensure the portions of the product they develop meet the team’s QA standards, including examples such as minimal defects/KLOC, runnable code with minimal crashing, and minimizing risk. This will include repeated testing of new methods, classes, and interfaces. Code Developers will report known defects and bugs to the Testing Engineers and together, potential solutions will be brainstormed. Also, to help the Testing Engineers be more efficient when completing unit tests, Code Developers will briefly explain how each function of the product should behave (such as a user being logged in after registering a new account).

The Testing Engineers will develop the team’s QA standards along with the Project Manager, who must then approve it if he deems the standards reasonable and doable.

3.4 QA Estimated Resources

The estimated resources required for QA on the University Website project are as follows:

* One Project Manager working full time for the entirety of the project.
* Two Requirement Engineers working full time for the entirety of the project.
* One Software Architect working full time for the entirety of the project.
* Two Integration Engineers working full time for the entirety of the project.
* Two Testing Engineers working full time for the entirety of the project.
* Eight Code Developers working full time (or as requested by the Project Manager) for the entirety of the project.

**4 Documentation**

4.1 Purpose

The purpose of this section is to identify the documentation that will be used to ensure team cohesion, product quality, and process efficiency.

4.2 Minimum Documentation Requirements

The following documents have been or will be produced:

* Software Quality Assurance Plan (SQAP)
* Software Configuration Management Plan (SCMP)
* Software Project Management Plan (SPMP)
* Software Requirements Specifications (SRS)
* Software Design Document (SDD)
* Software Verification and Validation Plan (SVVP)
* Any user documentation (such as manuals, guides, etc)

**5 Standards, Practices, Conventions, and Metrics**

5.1 Purpose

This section will describe the standards, practices, conventions, and metrics to be used and maintained for our University Website project. These are intended to ensure that levels of quality, consistency, and efficiency are met by all team members. Metrics will be important in measuring this quality, consistency, and efficiency.

5.2 Content

*Standards:*

The IEEE documentation standards will be used and maintained throughout all of the documentation contained in this project. The respective industry standards will be followed for the uses of UML, Java, scripting and markup languages, style sheets, database management, and version control.

*Practices:*

As previously mentioned, developers are encouraged to maintain quality standards while developing the product. As a result, it will lead to a more productive QA process as most defects would be taken care of during the development lifecycle.

All known defects will be shared at team meetings or with the Testing Engineers. Defects will be analyzed and potential solutions will be brainstormed with the goal of assuring the highest product quality.

Using Git, team members will be able to pull certain parts of the product for the goal of developing or fixing these parts of the product. While code developers are responsible for maintaining quality standards and testing their contributions, Testing Engineers will provide additional QA practices to ensure that as the product evolves, quality standards are continued to be met.

*Metrics:*

Metrics will be collected to track and more easily observe our team’s quality assurance practices while our project evolves. Collecting these metrics will enable our project to track defects, manage risks, identify points of product failure, and allow for quick and effective resolution of potential issues (defects, bugs, crashes, etc.).

*Quality Goals:*

Specific goals will be supplied in later revisions of this document.

**6 Reviews and Audits**

6.1 Purpose

The purpose of this section is to define the necessary reviews and audits to be conducted and how they will be accomplished. Reviews and audits will help to continuously focus the team’s attention on the quality of the project as it develops.

6.2 Minimum requirements

*6.2.1 Software Requirements Reviews (SRR)*

The Requirements Engineer (RE) is responsible for drafting the Software Requirements Specifications (SRS). This document outlines the requirements and specifications which will help build and guide the team as the product is developed. As such, it is important that a review of this document (and what it outlines), is conducted by the entire team while led by the RE and Project Manager.

*6.2.2 Preliminary Design Review (PDR)*

The preliminary design (also known as the top-level design) encompasses the basic architectural and abstract design of the product this team is building. The Unified Modeling Language (UML) is one such tool we will use to design the scope, relationships, and architecture of our product. A review of these foundational designs will be conducted by the team. Alternative designs will be discussed and one must be approved before any development cycles are scheduled to start.

*6.2.3 Critical Design Review (CDR)*

The Critical Design Review (also known as the Detailed Design) encompasses the specific details, functions, and behaviors our of product. This design, along with the preliminary design, will be outlined and diagramed by the Software Architects (SA) of the team. This design will be fragmented into detailed individual parts, allowing the team to observe how the entire project comes together. As such, proposed designs will be reviewed and approved by the entire team.

*6.2.4 Software Verification and Validation Plan Review (SVVPR)*

Team members responsible for QA in this project (i.e., Testing Engineers) are expected to write up the Software Verification and Validation Plan (SVVP), which is a staple in assuring our product holds up to the quality standards set by the entire team. A review of this document will be conducted by the Project Manager to approve the standards set by the SVVP.

*6.2.5 Functional Audit*

QA members will conduct audits of the product to ensure it follows the standards set by the SRS. The Requirements Engineer will guide the QA members in this audit.

*6.2.6 Physical Audit*

Prior to product delivery, QA members are responsible for ensuring that the physical software and its documentation (such as manuals, guides, etc.) are complete. A final approval must be granted by the Project Manager.

*6.2.7 In-Process Audits*

Team members will be responsible for verifying that their implementation is consistent with the approved design. To coordinate this, the Project Manager (PM) will perform audits of their work which could consist of questioning, debriefing, and presentations. As a result, the PM will be able to observe and document the progress of the team.

*6.2.8 Managerial Review*

Managerial reviews must be conducted by an organization element independent of the unit being reviewed or by a qualified third party. Since this is a student project directed by another student (who is the Project Manager), no viable element or party could be used for this review. Instead, the Project Manager will conduct in-process audits to track the team’s progress and use that for team reviews.

*6.2.9 Software Configuration Management Plan Review (SCMPR)*

The Software Configuration Management Plan (SCMP) is documented by the Integration Engineers of this team. The Project Manager will review the status of configuration management on a frequent basis.

*6.2.10 Post-Mortem Review*

At the conclusion of the project, team members will conduct reviews to assess the overall team performance and the Project Manager will provide a final review which will illustrate potential process improvement ideas.

*6.3 Other*

Intentionally left blank. During future revisions of this document, additional reviews or audits may be provided.

**7 Test**

Please refer to Section 3.3 for the QA/testing responsibilities that were described. The Testing Engineers may write up Software Test Documentation that will include details for testing our project.

**8 Problem Reporting and Corrective Action**

In our project, defects can arise from our documentation, database, or code.

In our code, we will measure defects according to the following values:

* Critical: Causes application to crash
* Serious: Causes at least one requirement or design feature to be unmet
* Trivial: Could be allowed to be left alone without impeding the user from using our website with ease
* Medium: Neither serious nor trivial.

Defects in the code can arise from syntax, logic, or data errors. As previously described in Section 3, known defects will be shared at team meetings. With the help of the Testing Engineers, team members will brainstorm potential solutions for these defects. The Project Manager (PM) will coordinate team members in defect resolution by approving solutions and assigning specific tasks.

Documentation defects can arise from improperly outlined documents. Therefore, continuous revisions are expected to occur and all team members will provide critical feedback for improving the documents. Final approval will be granted by the PM.

Defects in the project’s database management system will be approached with great caution when resolving them so as to not corrupt any pre-existing data. If a severe defect or issue does arise, a rollback to a recent backup will be utilized.

**9 Tools, Techniques, and Methodologies**

Software QA techniques include standards auditing, requirements tracing, design verification, and software reviews, inspections, and tests. Software QA tools are composed of project management programs which help visualize the entire project such as tasks, completed assignments, checklists, etc.

**10 Code Control**

Our project will be maintained and stored on Github. This includes all documentation, code, and database backups. For more information, please refer to the SCMP and SPMP.

**11 Media Control**

Not relevant to this project. All media, if any, will be maintained and stored on the project’s Github repository where every team member access to it.

**12 Supplier Control**

Not currently relevant.

**13 Records Collection, Maintenance, and Retention**

Team meeting minutes are recorded and stored in the project’s Github repository. Tasks and assignments are presented in Trello, a project management system we are utilizing for project visualization.

Besides those specifics, all records regarding the project will be stored on Github so every team member can access them.

**14 Training**

If any training is required for a team member regarding software development, they will personally be responsible to acquire that training. Examples could include training in web design, database management, or software design patterns.

At team meetings, potential orientations can be conducted, if necessary, regarding quality goals, tools, and techniques.

**15 Risk Management**

Team members of this project are encouraged to identify risks as early as possible and direct them to the Project Manager (PM). The PM will follow through with the appropriate risk management actions.

Identifying risks as early as possible will ensure the product evolves smoothly, because team members will be able to manage these risks immediately (through conquest or avoidance) and prevent major issues from arising in the project’s future.

**Appendix**

For future use.