SE-306 Software Quality Engineering



**Assignment # 3**

Title: Quality Assurance plan

Submitted By:

Muhammad Memoon (20-SE-030)

Sabha Rani (20-SE-032)

Mohammad Huzaifa (20-SE-045)

Submitted To:

# **Ma’am Farwa Aizaz**

# HITEC UNIVERSITY, TAXILA

Contents

[1. Purpose 3](#_Toc135653659)

[2. Reference 3](#_Toc135653660)

[3. Management 3](#_Toc135653661)

[4. Documentation 5](#_Toc135653662)

[5. Standards, Practices, Conventions, and Metrics 6](#_Toc135653663)

[6. Reviews and Audits 6](#_Toc135653664)

[7. Problem Reporting Procedures 6](#_Toc135653665)

[8. Tools, Techniques, and Methodologies 6](#_Toc135653666)

[9. Media Control 7](#_Toc135653667)

[10. Version Control 7](#_Toc135653668)

[11. Training 7](#_Toc135653669)

[12. Risks and Contingencies 7](#_Toc135653670)

[13. Schedule 8](#_Toc135653671)

[14. Approvals 9](#_Toc135653672)

[15. APPENDIX A. LIST OF ACRONYMS 9](#_Toc135653673)

**Quality Assurance Plan**

1. **Purpose**

The purpose of this document is to specify the software quality assurance plan for the development life cycle of the project "Online Cafeteria Order System." The objective of this software project is to create an online platform that allows users to place food orders from a cafeteria and facilitates efficient order management. This document follows the guidelines of the IEEE standard for Software Quality Assurance Plan, IEEE Std 730.1-1995. It is intended to fulfill the requirements for the completion of the Master of Software Engineering Project's Portfolio. The document will undergo review and evaluation by the major professor and the supervisory committee.

1. **Reference**

* Project Overview version 1.0,
* Software Requirements Specification, [Hitec University], [May, 2022],
* IEEE Guide for Software Quality Assurance Planning
* IEEE Standard for Software Quality Assurance Planning, IEEE Std 730.1-1995
* Software Project Management: A Unified Framework

1. **Management**
   1. **. Organization**

The project team consists of the following individuals:

**Project Manager:**

Memoon

**Major Professor:**

Ma’am Farwa Aizaz

**Software Developer(s):**

Memoon

**Quality Assurance Lead:**

Sabha Rani

**Formal Technical Inspectors**

Huzaifa

**3.2. Responsibilities**

**3.2.1 Project Manager**

The Project Manager is responsible for overseeing the entire project, including the quality assurance process. Their primary responsibilities include:

* Ensuring adherence to the software quality assurance plan
* Coordinating with the development team and stakeholders
* Monitoring project progress and addressing any quality-related issues

**3.2.2 Major Professor**

In addition to being a committee member, the major professor will supervise and evaluate all artifacts submitted by the developer. Reviews and walkthroughs of related materials will be conducted on a weekly basis.

**3.2.3 Software Developer(s)**

The Software Developer(s) is responsible for implementing the software and ensuring its quality. Their primary responsibilities include:

Following coding standards and best practices

* Conducting unit testing and ensuring code reliability
* Collaborating with the Quality Assurance Lead to address any defects or issues

**3.2.4 Quality Assurance Lead**

The Quality Assurance Lead is responsible for overseeing the quality assurance activities. Their primary responsibilities include:

* Reviewing project deliverables at different stages
* Conducting testing activities, including functional, integration, and system testing
* Documenting and reporting any defects or issues

**3.2.5 Formal Technical Inspectors**

The formal technical inspectors will provide formal reports on their inspection results for architecture design artifacts produced by the developer. Additionally, the following tasks will be conducted to ensure quality assurance:

1. **Requirements Analysis:** The developer should ensure that the software requirements specification clearly defines the functionality of the software and unambiguously declares the requirements that must be satisfied. The scope descriptions should outline what the software will allow and not allow.
2. **Design:** The developer and major professor will conduct reviews and analyses of the software's construction. They will discuss and scrutinize the strengths and weaknesses of various design techniques.
3. **Implementation:** The developer will conduct informal code reviews on a regular basis to ensure consistency with the design and identify any errors. JavaDoc will be produced to enhance maintainability and future work.
4. **Testing:** The developer will conduct tests as defined in the Software Test Plan to ensure the satisfaction of requirements and the reliability of the software.
5. **Documentation**

The following documentation will be generated and updated throughout the software life cycle:

**Phase I:**

1. **Vision Document:** Provides a detailed description of the project, including goals, constraints, and software requirements.
2. **Project Plan:** Illustrates major milestones and provides a timeline and effort estimation for the project.
3. **Software Quality Assurance Plan:** Outlines the plan for software quality assurance.

**Phase II:**

1. **Formal Requirement Specification:** Specifies the requirements using appropriate methodologies (e.g., UML/OCL).
2. **Test Plan:** Describes the test cases and methodologies for testing the software.
3. **Architecture Design:** Presents the object model and use cases.
4. **Formal Technical Inspection:** Includes inspection reports conducted by formal technical inspectors.

**Phase III:**

1. **User Manual:** Provides instructions on how to use the online cafeteria order system.
2. **Final Source Code:** Contains the complete source code of the system.
3. **Assessment Evaluation:** Evaluates the quality and success of the system.
4. **Project Evaluation:** Assesses the overall project and its contributions.
5. **Standards, Practices, Conventions, and Metrics**

**5.1 Coding Standards**

The following coding standards and practices will be followed:

* Programming Language - Java 8
* Coding Standard - Oracle Java Coding Conventions
* Code Documentation - JavaDoc
* Code Version Control - Git

**5.2 Testing Standards**

The following testing standards and practices will be followed:

* Test Framework - JUnit 5
* Test Documentation - IEEE Standard for Software Test Documentation

**5.3 Metrics**

The following metrics will be used:

* Source Lines of Code (SLOC) - To measure the size and complexity of the software.
* COCOMO I Model - To estimate the cost, effort, and duration of the project.

1. **Reviews and Audits**

Reviews and audits will be conducted throughout the software development process. Formal technical inspections will be performed on the architecture design document, and committee members will review and provide feedback on all produced documentation during presentations at the end of each phase.

1. **Problem Reporting Procedures**

The software developer should promptly report any problems encountered during the development process to the major professor. The major professor will assist in resolving the issues or, if necessary, escalate them to the supervisory committee for further discussion and resolution.

1. **Tools, Techniques, and Methodologies**

The following tools, techniques, and methodologies will be used throughout the software development process:

* **Development Environment:** Integrated Development Environment (IDE), such as Eclipse or IntelliJ IDEA.
* **Version Control:** Git for source code management and collaboration.
* **Collaboration Platform:** Communication and collaboration tools like Slack, Trello, or Microsoft Teams.
* **Testing Framework:** JUnit for unit testing.
* **Use Case Modeling:** UML (Unified Modeling Language) for modeling use cases and scenarios.
* **Requirements Specification:** UML and OCL (Object Constraint Language) for specifying requirements.
* **Software Design:** UML for designing the object model and class diagrams.
* **Documentation:** LaTeX or Microsoft Word for document preparation.
* **Project Management:** Agile methodologies (e.g., Scrum or Kanban) for project planning and tracking.

1. **Media Control**

Project documentation, including project plans, software quality assurance plans, and final reports, will be made available on a website or online repository accessible to the major professor and supervisory committee. Additionally, a CD will be created, containing all project documents and the final product.

1. **Version Control**

Version control will be employed to manage changes and track the progress of the project. The major professor or committee members will approve and increment version numbers for significant milestones. Different versions, such as alpha, beta, and the final version, will be released based on passing various testing stages and receiving approval.

1. **Training**

The software developer has received training in relevant courses such as software engineering, software management, software specifications, and software quality assurance. In addition to the formal education, the developer will also undertake self-study and research to enhance their knowledge and skills in software development, quality assurance, and relevant technologies. They will stay updated with the latest industry practices and trends by attending workshops, seminars, and conferences related to software engineering and quality assurance.

1. **Risks and Contingencies**

The following risks have been identified for the project:

**12.1. Technical Risks:**

* Integration challenges with existing systems
* Performance issues due to high user traffic
* Security vulnerabilities and data breaches

**12.2. Schedule Risks:**

* Delays in development due to unforeseen technical issues or dependencies
* Insufficient time allocation for testing and debugging

**12.3. Resource Risks:**

* Limited availability of skilled developers
* Inadequate hardware or software resources
* To mitigate these risks, the following contingency measures will be implemented:
* Regular monitoring and evaluation of technical risks to identify and address them early on.
* Continuous performance testing and optimization to ensure scalability and responsiveness.
* Implementation of security best practices, such as encryption and secure coding guidelines.
* Frequent project status updates and close coordination with stakeholders to manage schedule risks.
* Adequate resource allocation and timely procurement of necessary hardware and software resources.

1. **Schedule**

The project will follow a phased approach with predefined milestones and deliverables. The estimated schedule for the project is as follows:

**Phase I:** Project Initiation and Requirements Gathering (4 weeks)

* Vision Document
* Project Plan
* Software Quality Assurance Plan

**Phase II:** Design and Development (12 weeks)

* Formal Requirement Specification
* Test Plan
* Architecture Design
* Formal Technical Inspection

**Phase III:** Implementation, Testing, and Deployment (8 weeks)

* User Manual
* Final Source Code
* Assessment Evaluation
* Project Evaluation

Please note that the schedule is subject to change based on the complexity of the project, unforeseen circumstances, and resource availability.

1. **Approvals**

This Software Quality Assurance Plan requires approval from the major professor and supervisory committee. Any changes or updates to the plan should be reviewed and approved by the same parties.

1. APPENDIX A. LIST OF ACRONYMS

SQA: Software Quality Assurance

IEEE: Institute of Electrical and Electronics Engineers

MSE: Master of Software Engineering

UML: Unified Modeling Language

OCL: Object Constraint Language

IDE: Integrated Development Environment

Git: Distributed Version Control System

API: Application Programming Interface

HTML: Hypertext Markup Language

CSS: Cascading Style Sheets

SQL: Structured Query Language

JVM: Java Virtual Machine

UI: User Interface

UX: User Experience

CD: Compact Disc

HTTP: Hypertext Transfer Protocol

HTTPS: Hypertext Transfer Protocol Secure

IP: Internet Protocol

TCP: Transmission Control Protocol

UI: User Interface

UX: User Experience

CPU: Central Processing Unit

RAM: Random Access Memory

OS: Operating System

GUI: Graphical User Interface

SDK: Software Development Kit

API: Application Programming Interface

XML: Extensible Markup Language

JSON: JavaScript Object Notation

JVM: Java Virtual Machine

REST: Representational State Transfer

URL: Uniform Resource Locator

CRUD: Create, Read, Update, Delete

------------------------------------------------------------------------------------------------