**MEWAR INTERNATIONAL UNIVERSITY**

KM 21, ABUJA-KEFFI ROAD, NASARAWA STATE, NIGERIA

Education for Knowledge, Peace and Prosperity

Website:[https://miu.edu.ng](https://miu.edu.ng/), Email Address: [csd@miu.edu.ng](mailto:csd@miu.edu.ng)

**Department of Software Engineering**

**SEN303 Software Testing &Quality Assurance**

**Practical Log Book**  
 **For 300 Level Students**



**Prepared by: Department of Software Engineering**

**Approved by**: University Management

**Date: August 2021**

**Student Information**

**Name**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Matriculation Number**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Session**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Semester**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Instructions**

This logbook is for documenting all your practical activities in SEN 303.

Each session must be signed by the instructor at the end of the practical.

Students must include a brief reflection on what they learned after each session.

Late submission of this log book may result in penalties.

**Week 1: Introduction to Software Quality Assurance (SQA)**

**Objective**:

* To introduce key concepts of software quality assurance, its importance, and the development of a quality culture.

**Activities**:

Importance of software quality, building a culture of quality.

1. **Practical Task**: Conduct a brief review of a software product to identify potential areas for quality improvement.
2. **Deliverable**: A report on areas where quality can be improved in the selected software.

**Tools**:

* **Microsoft Word** for documentation.
* **Trello** for tracking quality issues.

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 2: Avoidance of Errors and Quality Problems**

**Objective**:

* To understand error prevention techniques and how to avoid common quality problems in software development.

**Activities**:

Common errors in software development and best practices to avoid them.

1. **Practical Task**: Analyze a set of code and identify errors or vulnerabilities.
2. **Deliverable**: A report listing identified errors and suggested improvements.

**Tools**:

* **SonarQube** for static code analysis.

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 3: Inspections and Reviews**

**Objective**:

* To practice inspections and reviews as a method to maintain software quality.

**Activities**:

How to conduct code inspections and peer reviews.

1. **Practical Task**: Organize a peer review session to inspect code from a small project.
2. **Deliverable**: Peer review report outlining key findings and suggestions.

**Tools**:

* **GitHub** or **GitLab** for collaborative code review.

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 4: Testing Techniques - Unit Testing**

**Objective**:

* To apply unit testing techniques for verifying individual components.

**Activities**:

Introduction to unit testing and its role in verification.

1. **Practical Task**: Write unit tests for a small software module.
2. **Deliverable**: Unit test report showing test results and code coverage.

**Tools**:

* **JUnit** (Java), **pytest** (Python) for unit testing.

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 5: Testing Techniques - Integration Testing**

**Objective**:

* To understand and implement integration testing techniques.

**Activities**:

Integration testing principles and techniques.

1. **Practical Task**: Develop integration tests for combining multiple software modules.
2. **Deliverable**: Integration test results showing how well components work together.

**Tools**:

* **Selenium** for web-based integration tests.
* **TestNG** for automated testing (Java-based projects).

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 6: Testing Techniques - System Testing**

**Objective**:

* To perform system testing, ensuring the entire software system meets requirements.

**Activities**:

System testing and its importance in the software development lifecycle.

1. **Practical Task**: Develop system tests for an end-to-end software system.
2. **Deliverable**: System test report summarizing the results of end-to-end testing.

**Tools**:

* **Selenium WebDriver** for automated system testing.

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 7: Process Assurance vs. Product Assurance**

**Objective**:

* To differentiate between process and product assurance and understand their importance.

**Activities**:

Process assurance vs. product assurance and their impact on software quality.

1. **Practical Task**: Conduct an audit of the software development process used in a project.
2. **Deliverable**: An audit report highlighting process improvements to ensure better quality.

**Tools**:

* **Trello** for process tracking.

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 8: Quality Process Standards**

**Objective**:

* To understand and apply quality process standards such as ISO 9001 and CMMI in software projects.

**Activities**:

Introduction to software quality standards like ISO 9001 and CMMI.

1. **Practical Task**: Map an existing project to a relevant quality process standard.
2. **Deliverable**: Documentation showing compliance with selected quality standards.

**Tools**:

* **Microsoft Excel** or **Google Sheets** for process mapping and documentation.

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 9: Verification and Validation (V&V)**

**Objective**:

* To apply verification and validation techniques in software testing.

**Activities**:

Principles of verification and validation in software projects.

1. **Practical Task**: Perform verification and validation on a software module to check for correctness.
2. **Deliverable**: V&V report summarizing findings and results of tests.

**Tools**:

* **JUnit**, **TestNG**, or **pytest** for validation.
* **Trello** for tracking verification tasks.

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 10: Problem Analysis and Reporting**

**Objective**:

* To analyze problems encountered in software development and report them effectively.

**Activities**:

Root cause analysis and problem reporting techniques.

1. **Practical Task**: Conduct root cause analysis on a software issue and submit a problem report.
2. **Deliverable**: Problem report with root cause analysis and recommended actions.

**Tools**:

* **Root Cause Analysis Tools** such as **5 Whys** or **Fishbone Diagram** (Ishikawa).

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 11: Statistical Approaches to Quality Control**

**Objective**:

* To use statistical methods to monitor and control software quality.

**Activities**:

Introduction to statistical quality control in software testing ( Six Sigma).

1. **Practical Task**: Apply statistical methods to analyze test results (e.g., defect density, test coverage).
2. **Deliverable**: A statistical report analyzing the software quality metrics.

**Tools**:

* **Minitab** or **R** for statistical analysis.
* **Excel** or **Google Sheets** for plotting quality metrics.

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Week 12: Final Project - Quality Assurance Strategy**

**Objective**:

* To develop a complete quality assurance strategy for a software project.

**Activities**:

Review of all topics and how to integrate them into a cohesive QA strategy.

1. **Practical Task**: Develop a comprehensive quality assurance strategy for the project, covering all areas from requirements, testing, process assurance, and statistical monitoring.
2. **Deliverable**: A full QA strategy document for a software project, including test plans, standards, and verification and validation processes.

**Tools**:

* **Trello** for process management and task tracking.

**Observation**:

**Challenges Faced**:

**Instructor’s Comments**:

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_