SUPPLEMENTARY SCHEDULE MANAGEMENT SYSTEM

SOFTWARE QUALITY ASSURANCE

❖ Electric and Electronic Engineering, University of Khartoum, Khartoum, Sudan

❖ Date: 5 April 2023 Prepared by: Quality Assurance Team Leader: Mohammed Ashraf

Introduction:

Quality assurance consists of the auditing and reporting functions of management. The goal of quality assurance is to provide management with the data necessary to be informed about product quality, thereby gaining insight and confidence that product quality is meeting its goals. Software Quality Assurance encompasses a quality management approach, effective software engineering technology (methods and tools), formal technical reviews that are applied throughout the software process, a multitier testing strategy, control of software documentation and the changes made to it, a procedure to ensure compliance with software development standards (when applicable), and measurement and reporting mechanisms.

Software Quality Assurance (SQA):

Quality Assurance concept is mainly about inspecting the product and evaluating its quality near its completion or at various stages of production. A software product is its functionality and not its use; this “invisible” nature of software adds to the complications of assessing its quality. Factors affecting Software Quality:

i. Our team: Monzer Omer is the team leader, Mujtaba Mirghani is the development team leader, Mohammed Ahmed Gasim is the testing team leader, Ahmed Al-Siddig is the developer, and Mohammed Ashraf is the requirement engineer & quality assurance team leader.

ii. Requirements: Dr. Rania, who conducts the procedure manually, has provided the necessary information, so we have provided all the information in the form of an accurate and detailed picture in addition to an accurate explanation of the issue.

iii. The schedule: We actually started at late December, and Mariam, our supervisor, guided us through each step until we finished the project. The team worked extremely tirelessly to complete it, and we turned it in at the early April.

We will measure software quality by emphasizing on the following:

- Software requirements are the foundation from which quality is measured. Lack of conformance to requirements is lack of quality.
- Specified standards define a set of development criteria that guide the manner in which software is engineered. If the criteria are not followed, lack of quality will almost surely result.

• A set of implicit requirements often goes unmentioned (e.g., the desire for ease of use and good maintainability). If software conforms to its explicit requirements but fails to meet implicit requirements, software quality is suspect.

SQA Activities

1. Requirements review: Review the software requirements to ensure they are complete, accurate, and testable: as I mentioned above Dr. Rania, who conducts the procedure manually, has provided the necessary information, so we have provided all the information in the form of an accurate and detailed picture in addition to an accurate explanation of the issue.

2. Test planning: Develop a test plan that outlines the testing approach, test cases, and test data to be used:

We reviewed every single case of user input and created strategies to deal with these mistakes.

3. Test case development: Develop test cases that cover all functional and non-functional requirements of the software:

The user who entered incorrectly appears on a page with all the instructions and instructions that assist him in entering correctly and resolving the problem after we have analyzed all of the user's problems and created our own system to be error tolerant.

4. Test execution: Execute the test cases and document the results:

The test document results were useful to us and assisted us in correcting these mistakes.

5. Defect tracking: Track and manage defects found during testing, including documenting the defect, assigning it to the appropriate team member, and tracking its resolution:

No defects detected.

6. Regression testing: Conduct regression testing to ensure that changes made to the software do not impact existing functionality:

The tests were carried out so that they are not significantly affected by potential factors and the system will operate effectively as required.

7. Performance testing: testing ensure that the software meets performance requirements.

8. Security testing: Because the system is not linked to a database or network, no security tests are performed or required.

9. User acceptance testing: the software meets the needs of the end-users.

10. Code review: the code is well-structured, maintainable, and adheres to coding standards.

11. Documentation review:

software documentation is complete, accurate, and up to date.

12. Configuration management:

the software configuration changes are tracked and controlled.

13. Release management:

the software is deployment plan is correctly set and meets the needs of the end-users.

14. Continuous improvement: We are planning to continuously improve the software quality assurance process by identifying areas for improvement and implementing changes to address them.

Software Reviews

Software reviews are a "filter" for the software engineering process. That is, reviews are applied at various points during software development and serve to uncover errors and defects that can then be removed. Software reviews "purify" the software engineering activities that we have called analysis, design, and coding.

➢ Here's an example software review for a supplementary tables' exams optimization project:

- Purpose: The purpose of the software is to optimize the creation and management of supplementary tables for exams.

- Functionality: The software allows users to easily create, edit, and manage supplementary tables for exams. It should also provide features such as data validation, formatting, and sorting.

- User interface: The user interface is intuitive and easy to use, with clear navigation and labelling. It should also be responsive and accessible on different devices and screen sizes.

- Performance: The software is fast and responsive, with minimal lag or delay when performing tasks such as loading data or saving changes.

- Security: The software is secure and protect sensitive data such as student information and exam results. It should also have appropriate access controls to ensure that only authorized users can access and modify data.

- Compatibility: The software is compatible with different operating systems, browsers, and devices to ensure that it can be used by a wide range of users.

- Documentation: The software is clear and comprehensive documentation, including user manuals, technical specifications, and release notes.

- Testing: The software is thoroughly tested to ensure that it meets all functional and non-functional requirements. This should include unit testing, integration testing, and user acceptance testing.

- Maintenance: The software is easy to maintain and update, with clear version control and change management processes in place.

- Scalability: The software is scalable and able to handle increasing amounts of data and users as the project grows.

❖ Overall, the software should be reliable, efficient, and user-friendly, with a focus on meeting the needs of the end-users and optimizing the creation and management of supplementary tables for exams.