**Software Quality Assurance (SQA) Plan for Quality Project:**

**Part A: Software Configuration Management:**

**1. Evaluations to be performed:**

Review of software configuration management (SCM) plan to ensure it is comprehensive and aligns with project objectives

• Inspection of SCM tools and processes to ensure they are appropriate and effective

• Review of SCM records to ensure they accurately reflect the current state of the software

**2. Audits and reviews to be performed:**

• Periodic audits of SCM processes to ensure they are being followed correctly

• Reviews of SCM records during key milestones to ensure they accurately reflect the current state of the software

**3. Standards that are applicable to the project:**

• ISO/IEC 15504 (SPICE)

•IEEE 828 (Standard for Software Configuration Management Plans)

**4. Procedures for error reporting and tracking:**

• All errors and issues discovered during SQ activities should be reported using the project's bug tracking tool

• Error reports should include a clear description of the issue, steps to reproduce it, and the expected outcome

• The SQA team will follow up on all error reports to ensure they are properly addressed .

**5. Documents to be produced by the SQA group:**

• Software Configuration Management Plan

SCM records (e.g. configuration management database, version control repository)

• Reports on the effectiveness of SCM processes and tools

**6. Amount of feedback provided to the software project team:**

• The SQA team will provide regular feedback to the software project team on the effectiveness of

SCM processes and tools

• Feedback will be provided during regular meetings and through written reports as needed

**Part B: To demonstrate the use of master control tools in performing software quality assurance**

**(SQA) activities, I will outline a simple process that could be followed:**

**1.** Identify the software to be tested and gather any relevant requirements or specifications.

**2.** Develop a test plan, which outlines the overall strategy for testing the software, including the types of tests that will be performed and the criteria that will be used to determine whether the software is functioning correctly.

**3.** Use a version control system (such as Git) to manage the code base and track changes as the software is developed. This will allow you to easily revert to previous versions of the code if necessary and ensure that all changes are properly documented.

**4.** As the software is developed, use a bug tracking tool (such as Jira) to document and track any issues that are identified. This will allow you to prioritize and address problems as they are found, and ensure that all issues are properly resolved before the software is released.

**5.** Use a static analysis tool (such as Coverity) to automatically scan the code for common coding errors and vulnerabilities. This can help to identify potential problems early in the development process, allowing you to fix them before they become more serious issues.

**6.** Perform manual and automated testing to validate that the software is functioning correctly and meets the specified requirements.

**7.** Use a continuous Integration (CI) system (such as Jenkins) to automate the build and test process, allowing you to quickly validate changes to the code and ensure that the software remains stable as it is cleveleped,

**8.** Use a code review teal (such as Review Board) to facilitate the review of code changes by other team members, ensuring that all changes are properly reviewed and approved before they are merged into the code base.

**9.** Monitor the software after it has been released to ensure that it continues to function correctly and that any issues that are identified are promptly addressed.

Overall, the use of master control tools can help to streamline the SQA process and ensure that the software meets the necessary quality standards.

Change control: Master control includes a change control module that allows users to submit, review, and track changes to the project or product. This is useful for ensuring that all changes are properly documented and that the impact of the change on the overall quality of the project or product is assessed.

**Track defect:** Master control also includes a defect tracking module that allows users to report and track defects in the project or product. This is useful for ensuring that defects are promptly addressed and that the root cause of the defect is identified and addressed to prevent future occurrences.

**Audit:** Master control includes an audit module that allows users to plan, conduct, and report on audits of the project or product. This is useful for ensuring that the project or product complies with relevant quality standards and regulations.

**Performance reporting:** Master control includes a performance reporting module that allows users to track and report on the performance of the project or product. This is useful for identifying areas for improvement and for demonstrating the quality of the project or product to stakeholders.

**here is a possible demo that showcases the use of Master control tools for performing various SQA activities in a project or product lifecycle:**

**1. Change control:** The demo begins with an overview of the project and its quality specifications. The team then demonstrates how they use Master control’s change control tool to manage and track changes to the project, including submitting and reviewing change requests, and ensuring that all changes are properly documented and tested before being implemented.

**2. Track defect:** The team then shows how they use Master control’s defect tracking tool to identify, document, and prioritize defects in the project. They demonstrate how the tool helps them assign defects to the appropriate team members, monitor their progress, and provide status updates to stakeholders.

**3. Audit:** The team then demonstrates how they use Master control’s audit tool to conduct regular audits of the project to ensure compliance with quality standards. They show how the tool helps them identify areas for improvement, document findings, and track the implementation of corrective actions.

**4. Performance reporting:** Finally, the team shows how they use Master control’s performance reporting tool to generate comprehensive reports on the project's progress and performance. They demonstrate how the tool allows them to track key metrics, such as defect rates, test coverage, and delivery schedules, and share the reports with stakeholders.

**To make a demo using master control tools, you will need to follow these steps:**

**1.** Determine the purpose of your demo. This will help you decide which tools to use and what content to include.

**2.** Choose the appropriate master control tools for your demo. Some common tools include switchers, character generators, audio mixers, and video servers.

**3.** Set up your equipment and test all of the connections to ensure that everything is working properly.

**4.** Create or gather the content that you will be using in your demo. This may include video clips, audio tracks, and graphics.

**5.** Use the master control tools to mix and switch between the different elements of your demo. For example, you may use a switcher to switch between video clips or a character generator to add titles and graphics to the screen.

**6.** Practice your demo to ensure that everything runs smoothly and that the transitions between elements are seamless.

**7.** Present your demo to your audience, whether it be in person or online.

I cant make a demo because the website is not working in Jordan :

