DevOps Engineer Assignment: Constructing a Jenkins CI/CD Pipeline

Objective:

The goal of this assignment is to create a Jenkins pipeline that integrates various open-source tools to assess code coverage, code quality, cyclomatic complexity, and security vulnerabilities. You will be demonstrating your ability to create an effective CI/CD process that aids in maintaining high-quality code standards in a software development project.

Requirements:

1. **Jenkins Setup**:
    - Install and configure Jenkins on a virtual machine or use a cloud-based service.
    - Ensure Jenkins is configured with the necessary plugins for Git and any other SCM tools you decide to use.
2. **Source Code Management**:
    - Use Git as the source code management tool.
    - Configure Jenkins to pull code from a Git repository (GitHub, GitLab, etc.).
3. **Pipeline Creation**:
    - Create a Jenkinsfile that defines the pipeline stages.
    - The pipeline should be triggered on every commit to the main branch of the repository.
4. **Code Quality Checks**:
    - Integrate SonarQube to analyze code quality and technical debt.
    - Configure the pipeline to break if the code quality gates are not met.
5. **Code Coverage**:
    - Integrate a tool like JaCoCo (for Java applications) or another relevant tool based on the programming language used.
    - Ensure that the code coverage report is published and accessible through Jenkins.
6. **Cyclomatic Complexity**:
    - Integrate a tool that can calculate and report the cyclomatic complexity of the code.
    - Tools such as Lizard (for C/C++, Java, Python, etc.) can be used.
7. **Security Vulnerability Scan**:
    - Integrate OWASP Dependency-Check or another similar tool to scan for known security vulnerabilities in the project dependencies.
    - Ensure that the vulnerability reports are accessible and that builds fail if critical vulnerabilities are found.
8. **Notifications**:
    - Configure Jenkins to send notifications (email, Slack, etc.) on build success or failure.
9. **Documentation**:

    - Document the pipeline steps, tools integrated, and any configurations needed for the setup.
    - Include troubleshooting steps for common issues that might occur during the build process.

Deliverables:

1. Jenkinsfile with complete pipeline configuration.
2. Documentation covering setup, configuration, and usage of the pipeline.
3. A report outlining the results of the initial run of the pipeline, including screenshots of the dashboard of each tool used.

Evaluation Criteria:

- Completeness of the pipeline setup.
- Ability to integrate multiple tools effectively.
- Clarity and completeness of documentation.
- Handling of different scenarios (e.g., build failures due to quality checks).

Submission:

Submit your work through a GitHub repository link containing all the required files and documentation. Ensure that the repository is public or accessible to reviewers.