Q1. What is continuous monitoring in the context of CI/CD? Write a short note on either Prometheus or Nagios as a monitoring tool.

Sol: Continuous Monitoring is a key practice in DevOps and CI/CD pipelines that involves the continuous collection, analysis, and feedback of system metrics and logs to ensure the health, performance, and security of applications and infrastructure. It helps in early detection of issues, improves system reliability, and supports faster resolution through real-time insights.

Prometheus

- **Type**: Open-source monitoring and alerting toolkit
- **Purpose**: Collects metrics in real-time, stores them in a time-series database
- **Best For**: Monitoring cloud-native applications, Kubernetes, and containerized systems
- **Features**:
    - Pull-based model for collecting metrics
    - Built-in query language (PromQL)
    - Integration with Grafana for dashboards
    - Alertmanager for handling alerts
- **Use in CI/CD**: Tracks performance metrics after deployment, monitors system health continuously

---

Nagios

- **Type**: Open-source infrastructure monitoring tool
- **Purpose**: Monitors servers, network, and applications for availability and performance
- **Best For**: Traditional IT infrastructure monitoring (on-premise or hybrid)
- **Features**:
    - Alerting system for failures and recoveries

    - Plugin-based architecture for extending functionality
    - Dashboards and reports for system status
- **Use in CI/CD**: Ensures environment stability during and after deployments, detects downtime early

Q2. What is Continuous Feedback in a CI/CD pipeline, and why is it critical to the success of DevOps? Explain how feedback should be managed for maximum effectiveness, including characteristics it should have and avoid. Also, mention the rules teams should follow when handling feedback during development.

Sol: pages 37 to 40 of CICD_Module_CD.pdf

You are working on a software project that follows a Continuous Integration/Continuous Deployment (CI/CD) pipeline. The development team wants to ensure **automated testing and quality assurance** through the following tools.

1. JUnit
2. SonarQube
3. Snyk

A. Identify at which stages of the CI/CD pipeline the above tools are typically used. Provide the reason for using each tool at that specific stage and describe what kind of testing or analysis each performs.
B. Describe how the use of these tools contributes to software quality and security assurance in the pipeline.

C. Provide commonly used metrics or outputs generated by each tool that help developers make informed decisions.