CI/CD with Jenkins and SonarQube – Student Exercises

# Exercise 1 – Create a Simple Jenkins Pipeline

Goal: Understand basic pipeline structure and syntax  
  
Instructions:  
• Create a new folder in the project: `jenkins-demo`  
• Add a basic `Jenkinsfile` with one stage:  
  
 pipeline {  
 agent any  
 stages {  
 stage('Hello') {  
 steps {  
 echo 'Hello Jenkins!'  
 }  
 }  
 }  
 }  
  
• Push to GitHub and run the pipeline in Jenkins  
  
Outcome: Learn how Jenkins detects and runs a Jenkinsfile.

# Exercise 2 – Add Build and Test Stages

Goal: Automate build and unit testing  
  
Instructions:  
• Edit the pipeline to include:  
  
 stage('Build and Test') {  
 steps {  
 sh './mvnw clean verify'  
 }  
 }  
  
• Add a test report stage:  
  
 stage('Results') {  
 steps {  
 junit 'target/surefire-reports/\*.xml'  
 }  
 }  
  
Outcome: Understand automated testing and test reporting.

# Exercise 3 – Integrate SonarQube

Goal: Analyze code quality with SonarQube  
  
Instructions:  
• Add a new stage to run Sonar analysis:  
  
 stage('SonarQube Analysis') {  
 steps {  
 withSonarQubeEnv('SonarQube') {  
 sh "./mvnw sonar:sonar -Dsonar.projectKey=bookservice -Dsonar.login=${SONAR\_TOKEN}"  
 }  
 }  
 }  
  
• Trigger the pipeline and verify results in SonarQube dashboard  
  
Outcome: Learn how CI pipelines enforce code quality standards.

# Exercise 4 – (Optional) Introduce a Bug and Fix It

Goal: See how CI/CD catches issues early  
  
Instructions:  
• Comment out a test or introduce a failing one  
• Push the change and observe Jenkins + SonarQube reaction  
• Fix the bug and commit again  
  
Outcome: Reinforce the role of fast feedback and test-driven quality.

# Exercise 5 – (Optional) Analyze Coverage with JaCoCo

Goal: Visualize how code coverage is integrated  
  
Instructions:  
• Ensure `jacoco-maven-plugin` is in `pom.xml`  
• Run the pipeline  
• Open the report at `target/site/jacoco/index.html`  
• Discuss lines not covered and how to test them  
  
Outcome: Understand the relationship between tests and quality metrics.