```
pipeline {
    agent any
    tools {
        jdk 'jdk17'
        nodejs 'node16'
    }
    environment {
        SCANNER_HOME = tool 'sonar-scanner'
    }
    stages {
        stage('Clean Workspace') {
            steps {
                cleanWs()
            }
        }
        stage('Checkout from Git') {
            steps {
                git branch: 'main', url:
'https://github.com/Aj7Ay/Netflix-clone.git'
            }
        }
        stage('Sonarqube Analysis') {
            steps {
                withSonarQubeEnv('sonar-server') {
                    sh '''
                        $SCANNER_HOME/bin/sonar-scanner -
Dsonar.projectName=Netflix \
                        -Dsonar.projectKey=Netflix
                    '''
                }
            }
        }
        stage('Quality Gate') {
            steps {
                script {
                    waitForQualityGate abortPipeline: false,
credentialsId: 'sonar-token'
                }
            }
        }
        stage('Install Dependencies') {
            steps {
                sh "npm install"
            }
        }
        stage('OWASP FS Scan') {
            steps {
                dependencyCheck additionalArguments: '--scan ./ --
disableYarnAudit --disableNodeAudit', odcInstallation: 'DP-Check'
                dependencyCheckPublisher pattern: '**/dependency-check-
report.xml'
            }
        }
        stage('Trivy FS Scan') {
```

```
            steps {
                sh "trivy fs . > trivyfs.txt"
            }
        }
        stage('Docker Build & Push') {
            steps {
                script {
                    withDockerRegistry(credentialsId: 'docker', toolName:
'docker') {
                        sh "docker build --build-arg
TMDB_V3_API_KEY=<API_KEY> -t netflix ."
                        sh "docker tag netflix
tigerninja32/netflix:latest"
                        sh "docker push tigerninja32/netflix:latest"
                    }
                }
            }
        }
        stage('Trivy Image Scan') {
            steps {
                sh "trivy image tigerninja32/netflix:latest >
trivyimage.txt"
            }
        }
        stage('Deploy to Container') {
            steps {
                script {
                    sh '''
                    if [ $(docker ps -q -f name=netflix) ]; then
                        docker stop netflix
                        docker rm netflix
                    fi
                    docker run -d --name netflix -p 8081:80
tigerninja32/netflix:latest
                    '''
                }
            }
        }
        stage('Deploy to Kubernetes') {
            steps {
                script {
                    dir('Kubernetes') {
                        withKubeConfig(caCertificate: '', clusterName:
'', contextName: '', credentialsId: 'k8s', namespace: '',
restrictKubeConfigAccess: false, serverUrl: '') {
                            sh 'kubectl apply -f deployment.yml'
                            sh 'kubectl apply -f service.yml'
                        }
                    }
                }
            }
        }
    }
    post {
```

```
        always {
            emailext(
                attachLog: true,
                subject: "'${currentBuild.result}'",
                body: "Project: ${env.JOB_NAME}<br/>" +
                      "Build Number: ${env.BUILD_NUMBER}<br/>" +
                      "URL: ${env.BUILD_URL}<br/>",
                to: 'terrance.watkins54@gmail.com',
                attachmentsPattern: 'trivyfs.txt,trivyimage.txt'
            )
        }
    }
}
```