Jenkins Pipeline Script

```
pipeline {
    agent any

    tools {
        jdk 'jdk21'
        nodejs 'node-js'
    }

    environment {
        SCANNER_HOME = tool 'sonar'
        DOCKER_IMAGE = 'yashukl2002/bms:latest'
        EKS_CLUSTER_NAME = 'yashu-cluster'
        AWS_REGION = 'us-west-1'
    }

    stages {
        stage('Clean Workspace') {
            steps {
                cleanWs()
            }
        }

        stage('Checkout from Git') {
            steps {
                git branch: 'main', url: 'https://github.com/Yashutej/Book-My-Show-capstone.git'
                sh 'ls -la'  // Verify files after checkout
            }
        }

        stage('SonarQube Analysis') {
```

```
    steps {
        withSonarQubeEnv('sonar-server') {
            sh '''
            $SCANNER_HOME/bin/sonar-scanner \
                -Dsonar.projectName=BMS \
                -Dsonar.projectKey=BMS
            '''
        }
    }
}

stage('Quality Gate') {
    steps {
        script {
            // Added timeout to avoid infinite waiting
            timeout(time: 1, unit: 'MINUTES') {
                def qg = waitForQualityGate abortPipeline: true, credentialsId: 'sonar-token'
                echo "Quality Gate status: ${qg.status}"
            }
        }
    }
}

stage('Install Dependencies') {
    steps {
        sh '''
        cd bookmyshow-app
        ls -la  # Verify package.json exists
        if [ -f package.json ]; then
            rm -rf node_modules package-lock.json  # Remove old dependencies
```

```
                npm install  # Install fresh dependencies
            else
                echo "Error: package.json not found in bookmyshow-app!"
                exit 1
            fi
            '''
        }
    }

    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 '''
    if ! command -v trivy &> /dev/null
    then
        echo "Trivy not installed! Skipping scan."
        exit 0
    fi

    echo "Running Trivy FS scan..."
    trivy fs . > trivyfs.txt
    '''
  }
}
```

```
        stage('Docker Build & Push') {
            steps {
                script {
                    withDockerRegistry(credentialsId: 'docker', toolName: 'docker') {
                        sh '''
                        echo "Building Docker image..."
                        docker build --no-cache -t $DOCKER_IMAGE -f bookmyshow-app/Dockerfile bookmyshow-app

                        echo "Pushing Docker image to Docker Hub..."
                        docker push $DOCKER_IMAGE
                        '''
                    }
                }
            }
        }

        stage('Deploy to EKS Cluster') {
            steps {
                script {
                    sh '''
                    echo "Verifying AWS credentials..."
                    aws sts get-caller-identity

                    echo "Configuring kubectl for EKS cluster..."
                    aws eks update-kubeconfig --name $EKS_CLUSTER_NAME --region $AWS_REGION

                    echo "Verifying kubeconfig..."
```

```
                kubectl config view

                echo "Deploying application to EKS..."
                kubectl apply -f deployment.yml
                kubectl apply -f service.yml

                echo "Verifying deployment..."
                kubectl get pods
                kubectl get svc
                '''
            }
        }
    }

    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: 'klyashu632@gmail.com',
                attachmentsPattern: 'trivyfs.txt'
        }
    }
}
```

Docker File

```
# Use Node.js 18 (or your Jenkins-configured version)
FROM node:18

# Set working directory
WORKDIR /app

# Copy package.json and package-lock.json
COPY package.json package-lock.json ./

# Force install a compatible PostCSS version to fix the issue
RUN npm install postcss@8.4.21 postcss-safe-parser@6.0.0 --legacy-peer-deps

# Install dependencies
RUN npm install

# Copy the entire project
COPY . .

# Expose port 3000
EXPOSE 3000

# Set environment variable to prevent OpenSSL errors
ENV NODE_OPTIONS=--openssl-legacy-provider
ENV PORT=3000

# Start the application
CMD ["npm", "start"]
```

Deployment YAML File

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: bms-app
  namespace: yashaswini
  labels:
    app: bms
spec:
  replicas: 2
  selector:
    matchLabels:
      app: bms
  template:
    metadata:
      labels:
        app: bms
    spec:
      containers:
      - name: bms-container
        image: yashukl2002/bms:latest
        ports:
        - containerPort: 3000
```

Service YAML File

```
apiVersion: v1
kind: Service
metadata:
  name: bms-service
```

```
  namespace: yashaswini
  labels:
    app: bms
spec:
  type: LoadBalancer
  ports:
  - port: 80
    targetPort: 3000  # Replace with the port your app runs on
  selector:
    app: bms
```

Prometheus YAML file

```
global:
  scrape_interval: 15s


scrape_configs:
  - job_name: 'prometheus'
    static_configs:
      - targets: ['154.212.196.55:9090']


  - job_name: 'node-exporter'
    static_configs:
      - targets: ['54.212.196.55:9100']


  - job_name: 'jenkins'
    metrics_path: '/prometheus'
    static_configs:
      - targets: ['35.90.27.167:8080']
```