Jenkins CI/CD pipeline stages breakdown for deploying a Node.js application, especially when targeting **AWS EKS** or similar environments.

--------=----------=----------=----------=----------=----------=--

Pipeline Overview

The pipeline stages will look like this:

```
Stage 1 → Code Checkout
Stage 2 → Code Analysis (SonarQube)
Stage 3 → Install Dependencies
Stage 4 → Unit Testing (Jest/Mocha)
Stage 5 → Build Docker Image
Stage 6 → Push Docker Image to ECR
Stage 7 → Deploy to EKS (kubectl apply)
```

----------=---------=----------=----------=----------=--------=----------

Detailed Stage Breakdown

1. Code Checkout

- **Purpose:** Pull latest code from GitHub or GitLab repository.
- **Jenkinsfile Example:**

```
groovy
CopyEdit
stage('Checkout') {
    steps {
        git branch: 'main', url: 'https://github.com/your-repo/nodejs-app.git'
    }
}
```

----------=----------=----------=----------=----------=----------

2. Code Analysis (SonarQube)

- **Purpose:** Perform static code analysis for bugs, vulnerabilities, and code smells.
- **Jenkinsfile Example:**

```
groovy
```

```
CopyEdit
stage('SonarQube Analysis') {
    environment {
        scannerHome = tool 'SonarQubeScanner'
    }
    steps {
        withSonarQubeEnv('SonarQubeServer') {
            sh "${scannerHome}/bin/sonar-scanner \
                -Dsonar.projectKey=nodejs-app \
                -Dsonar.sources=. \
                -Dsonar.host.url=http://<sonarqube-url> \
                -Dsonar.login=<token>"
        }
    }
}
```

---

3. Install Dependencies

- **Purpose:** Install required npm packages.
- **Jenkinsfile Example:**

```
groovy
CopyEdit
stage('Install Dependencies') {
    steps {
        sh 'npm install'
    }
}
```

---

4. Unit Testing (Jest/Mocha)

- **Purpose:** Ensure code works as expected before building.
- **Jenkinsfile Example:**

```
groovy
CopyEdit
stage('Unit Tests') {
    steps {
```

```
        sh 'npm test'
    }
}
```

---

5. Build Docker Image

- **Purpose:** Containerize Node.js app.
- **Jenkinsfile Example:**

```
groovy
CopyEdit
stage('Build Docker Image') {
    steps {
        script {
            sh """
            docker build -t ${ECR_REPO}:${BUILD_NUMBER} .
            """
        }
    }
}
```

---

6. Push Docker Image to ECR

- **Purpose:** Store image in AWS Elastic Container Registry.
- **Jenkinsfile Example:**

```
groovy
CopyEdit
stage('Push to ECR') {
    steps {
        script {
            sh """
            aws ecr get-login-password --region ${AWS_REGION} \
            | docker login --username AWS --password-stdin
${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_REGION}.amazonaws.com
```

```
            docker tag ${ECR_REPO}:${BUILD_NUMBER}
${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_REGION}.amazonaws.com/${ECR_REPO}:${BU
ILD_NUMBER}
            docker push
${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_REGION}.amazonaws.com/${ECR_REPO}:${BU
ILD_NUMBER}
            """
        }
    }
}
-------------=------------------------=---------------------=------------------=-----------=---------
```

7. Deploy to EKS

- **Purpose:** Apply Kubernetes manifests to EKS.
- **Jenkinsfile Example:**

```
groovy
CopyEdit
stage('Deploy to EKS') {
    steps {
        script {
            sh """
            aws eks update-kubeconfig --name ${EKS_CLUSTER} --region
${AWS_REGION}
            kubectl apply -f k8s/deployment.yaml
            kubectl apply -f k8s/service.yaml
            """
        }
    }
}
--------------=------------------=-----------------=--------------=------------=-----------=--------
```