Here's a **very simple Jenkins Declarative Pipeline** with build + SonarQube quality gate 👍

```
pipeline {
  agent any

  stages {

    stage('Checkout') {
      steps {
        git 'https://github.com/sample/repo.git'
      }
    }

    stage('Build') {
      steps {
        sh 'mvn clean package'
      }
    }

    stage('SonarQube Scan') {
      steps {
        withSonarQubeEnv('SonarQube') {
          sh 'mvn sonar:sonar'
        }
      }
    }

    stage('Quality Gate') {
      steps {
        waitForQualityGate abortPipeline: true
      }
    }
  }
}
```

## 🧠 What this pipeline does (simple):

- Download code
- Build using Maven
- Send code to SonarQube
- Stop pipeline if quality gate fails

✅ **Interview one-liner:**

“This Jenkins pipeline builds code, runs SonarQube analysis, and stops automatically if the Quality Gate fails.”

## If you remove the Quality Gate step:

✔️ SonarQube will still **scan and review** the code

✔️ You can still see results in SonarQube dashboard

❌ But Jenkins will **not stop** the pipeline

👉 Even if code quality fails, build continues (deploy may still happen)

## Simple idea:

- SonarQube scan = analysis only
- Quality Gate step = enforce rules in pipeline

✅ **Interview one-liner:**

“If the Quality Gate step is removed, SonarQube still analyzes the code but Jenkins won’t fail the pipeline based on the result.”