Here is an example Jenkinsfile to create the pipeline:

```
pipeline {
 agent any

 stages {
  stage('Checkout') {
   steps {
    checkout([$class: 'GitSCM',
         branches: [[name: '*/master']],
         userRemoteConfigs: [[url: 'https://github.com/example/sample-app.git']]])
   }
  }

  stage('Code Build') {
   steps {
    sh './gradlew clean build'
   }
  }

  stage('Docker Build') {
   steps {
    sh 'docker build -t my-app .'
   }
  }

  stage('Publish') {
   steps {
    withCredentials([usernamePassword(credentialsId: 'dockerhub', usernameVariable:
'USERNAME', passwordVariable: 'PASSWORD')]) {
```

```
          sh "docker login -u $USERNAME -p $PASSWORD"
        }
        sh "docker tag my-app username/my-app:latest"
        sh "docker push username/my-app:latest"
      }
    }
  }
}
```

Explanation:

The agent any directive specifies that the pipeline should run on any available agent.

The Checkout stage uses the Git plugin to checkout the source code from the Git repository.

The Code Build stage runs a Gradle build to build the application.

The Docker Build stage builds a Docker image using the Dockerfile in the project directory.

The Publish stage logs into Docker Hub using the credentials stored in Jenkins, tags the Docker image, and pushes it to Docker Hub.

Note: Replace https://github.com/example/sample-app.git with your Git repository URL and username with your Docker Hub username.

This pipeline will automate the build, testing, and deployment process of your application every time you push changes to the Git repository.