trigger: # A push trigger specifies which branches cause a continuous integration build to run.

 branches:

 include: # Branch names to include for triggering a run.

 - dev

 - qa

 - main

pool: # The pool keyword specifies which pool to use for a job of the pipeline.

 vmImage: 'ubuntu-latest' # Name of the VM image you want to use

 demands: maven # List of demands (for a private pool).

steps:

 - task: DownloadSecureFile@1 # Download a secure file to the agent machine.

 displayName: 'Download a secure file to the agent machine'

 inputs:

 secureFile: settings.xml #Specifies the name or unique identifier (GUID) of the secure file that is downloaded to the agent machine.

 - task: Bash@3 # Run a Bash script on macOS, Linux, or Windows.

 displayName: 'Settings File Setup'

 inputs:

 targetType: 'inline' # Targets script type: file path or inline.

 script: | # Required when targetType = inline. The contents of the script.

 mkdir ~/.m2

 cp $AGENT\_TEMPDIRECTORY/settings.xml ~/.m2/settings.xml

 - task: Maven@3 # Use this task to build, test, and deploy with Apache Maven.

 inputs:

 mavenPomFile: 'pom.xml' # Specifies the relative path from the repository root to the Maven POM file

 publishJUnitResults: true # Publish the JUnit test results produced by the Maven build to Azure Pipelines

 javaHomeOption: 'JDKVersion' # Sets JAVA\_HOME either by selecting a JDK version that will be discovered during builds or by manually entering a JDK path.

 mavenOptions: '-DskipTests=false' # Sets the MAVEN\_OPTS environment variable, which is used to send command-line arguments to start the JVM.

 - task: CopyFiles@2 # Use this task to copy files from a source folder to a target folder using match patterns.

 inputs:

 Contents: '\*\*/\*.jar' #The file paths to include as part of the copy.

 TargetFolder: '$(Build.ArtifactStagingDirectory)' # The target folder or UNC path that will contain the copied files.

 CleanTargetFolder: true # Deletes all existing files in the target folder before the copy process.

 - task: PublishBuildArtifacts@1 # Use this task in a build pipeline to publish build artifacts to Azure Pipelines, TFS, or a file share.

 condition: and(succeeded(), eq(variables['Build.SourceBranch'], 'refs/heads/dev')) # Check is the current source branch is dev

 inputs:

 PathtoPublish: '$(Build.ArtifactStagingDirectory)' # Specifies the folder or file path to publish.

 ArtifactName: '{Artifact Name for DEV}' # Specifies the name of the artifact to create in the publish location.

 publishLocation: 'Container' # Store the artifact in Azure Pipelines (Container)

 displayName: Publish Artifact For Dev

 - task: PublishBuildArtifacts@1 # Use this task in a build pipeline to publish build artifacts to Azure Pipelines, TFS, or a file share.

 condition: and(succeeded(), eq(variables['Build.SourceBranch'], 'refs/heads/qa')) # Check is the current source branch is qa

 inputs:

 PathtoPublish: '$(Build.ArtifactStagingDirectory)' # Specifies the folder or file path to publish.

 ArtifactName: '{Artifact Name for QA}' # Specifies the name of the artifact to create in the publish location.

 publishLocation: 'Container' # Store the artifact in Azure Pipelines (Container)

 displayName: Publish Artifact For QA

 - task: PublishBuildArtifacts@1 # Use this task in a build pipeline to publish build artifacts to Azure Pipelines, TFS, or a file share.

 condition: and(succeeded(), eq(variables['Build.SourceBranch'], 'refs/heads/main')) # Check is the current source branch is main

 inputs:

 PathtoPublish: '$(Build.ArtifactStagingDirectory)' # Specifies the folder or file path to publish.

 ArtifactName: '{Artifact Name for PROD}' # Specifies the name of the artifact to create in the publish location.

 publishLocation: 'Container' # Store the artifact in Azure Pipelines (Container)

 displayName: Publish Artifact For Prod