Jenkins Scripted Pipeline (Groovy)

--> For scripted pipeline, we will be using Groovy language

--> Flexibility

--> Customization

--> Reusable components

Syntax:

node {

stage(‘git clone’){

echo ‘git cloning’

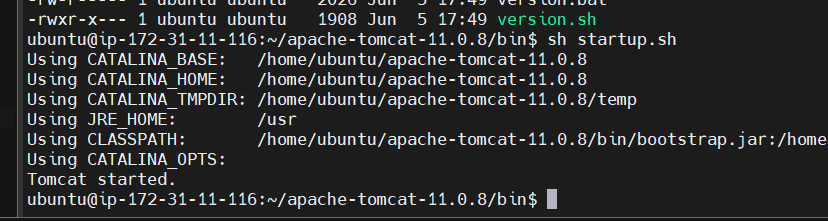
}

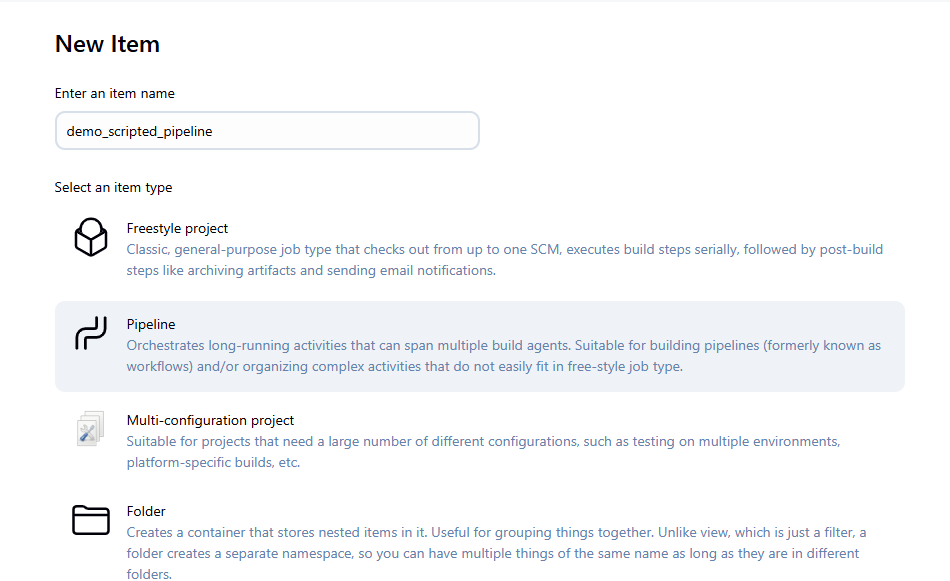
stage(‘build’){

echo ‘build…’

}

}







node {

stage('Cloning') {

echo 'git repo cloning'

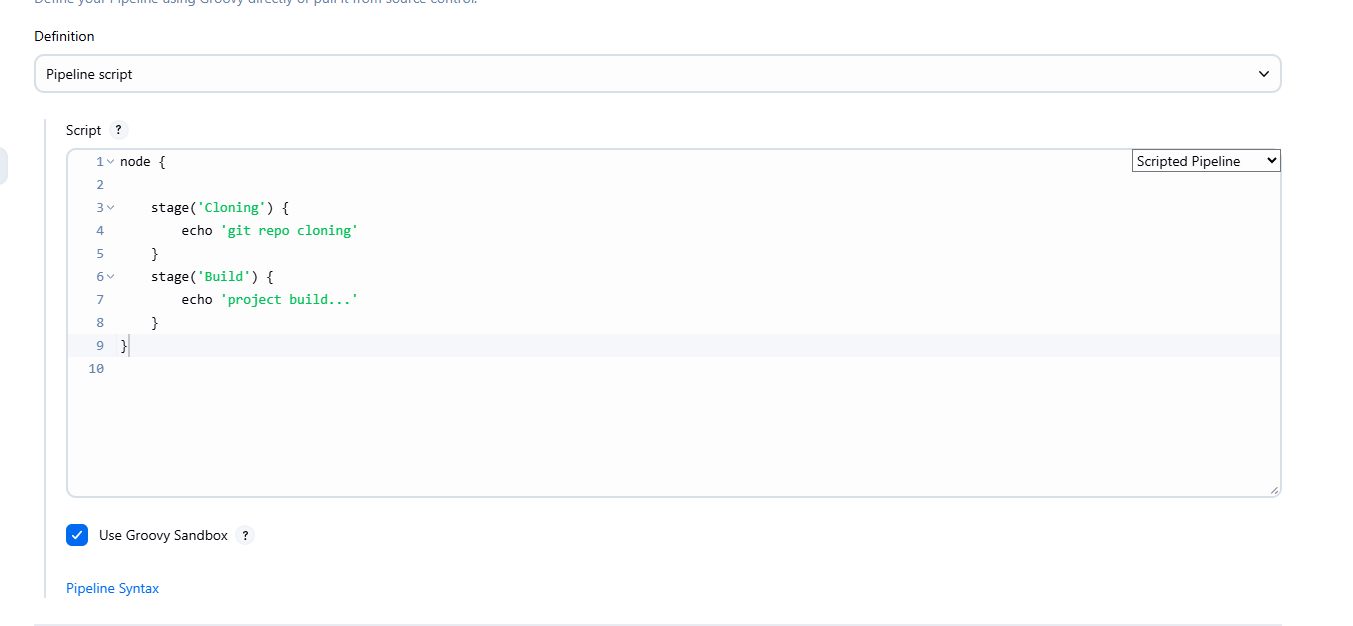
}

stage('Build') {

echo 'project build...'

}

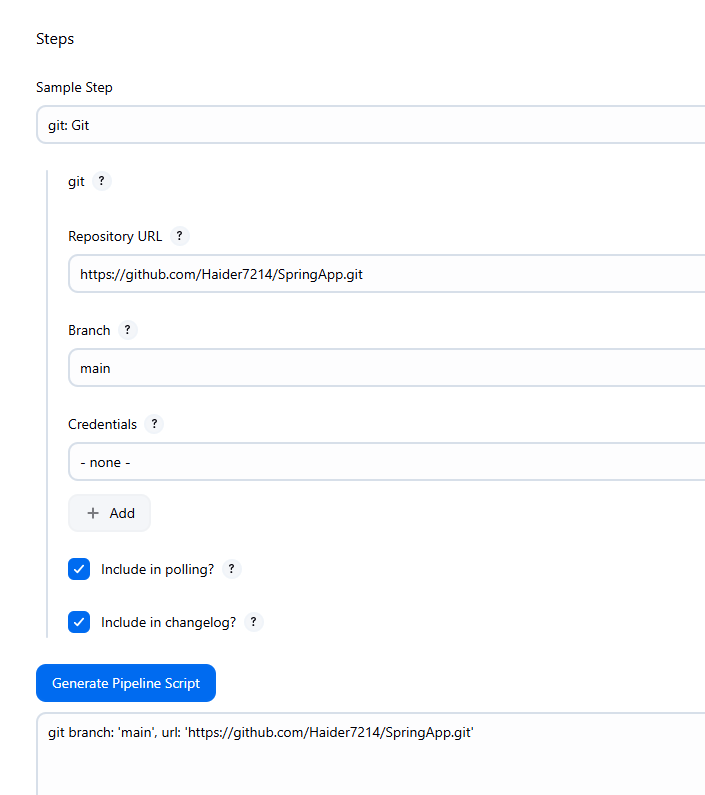
}



Apply and Save



Go back to Pipeline --> Configure



For Declarative it is different, for Scripted it is totally different

git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'

node {

def mvnPath

stage('git clone') {

git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'

}

stage('Maven Build') {

def mvnHome = tool name:'maven:3.9.10', type='maven';

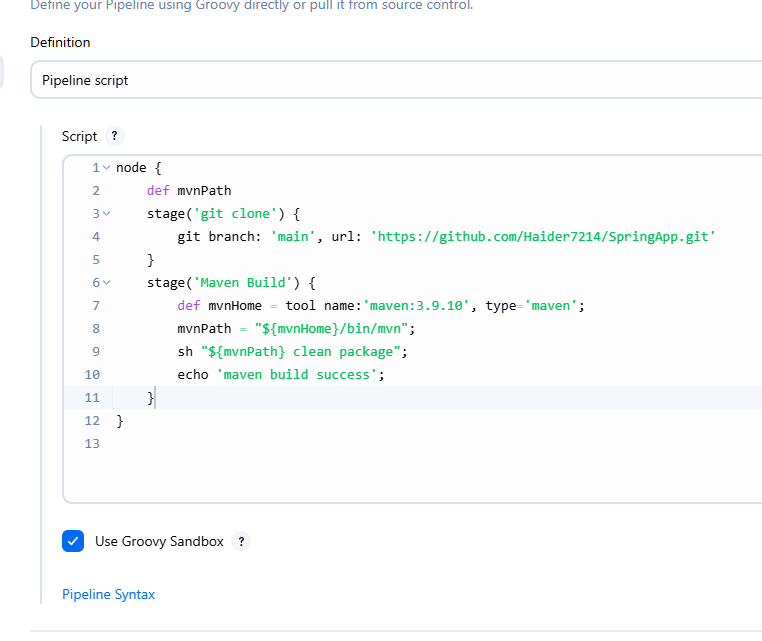
mvnPath = "${mvnHome}/bin/mvn";

sh "${mvnPath} clean package";

echo 'maven build success';

}

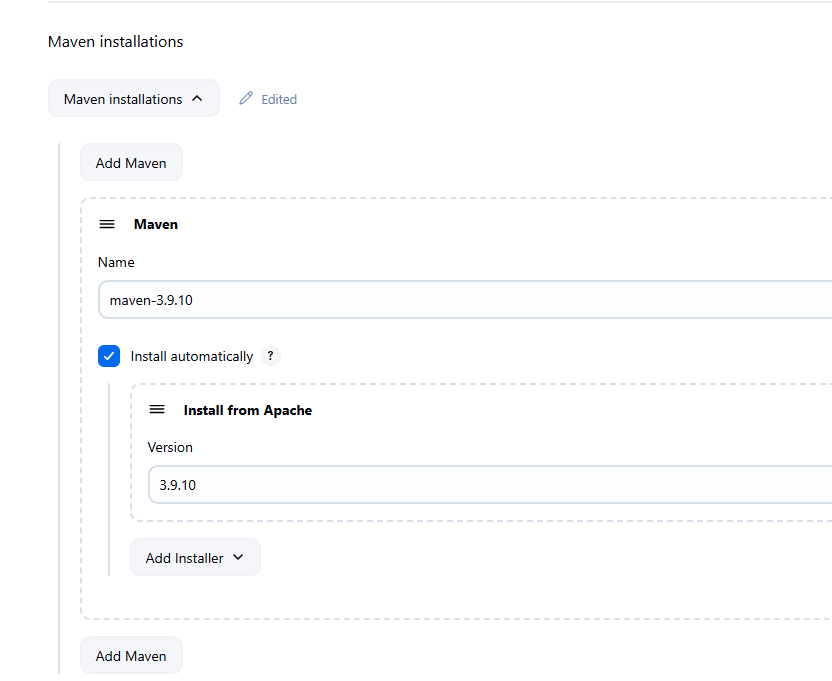
}



Apply and Save

Manage Jenkins --> Tools

That’s the name we have given for Maven



Failed



I change the script

node {

def mvnPath

stage('git clone') {

git branch: 'main', url: 'https://github.com/Haider7214/SpringBoot.git'

}

stage('Maven Build') {

def mvnHome = tool name:'maven-3.9.10', type='maven';

mvnPath = "${mvnHome}/bin/mvn";

sh "${mvnPath} clean package";

echo 'maven build success';

}

}

Again Build failed

node {

def mvnPath

stage('git clone') {

git branch: 'main', url: 'https://github.com/Haider7214/SpringBoot.git'

}

stage('Maven Build') {

def mvnHome=tool name:'maven-3.9.10', type:'maven';

mvnPath="${mvnHome}/bin/mvn";

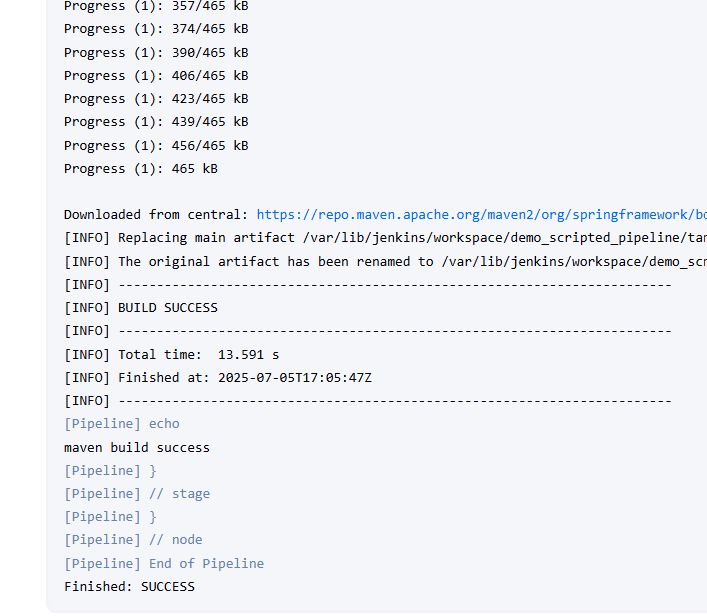
sh "${mvnPath} clean package";

echo 'maven build success';

}

}

This time build was successful



Jenkins Scripted (Groovy) Pipeline with Git + Maven

node {

def mvnPath

stage('git clone') {

git branch: 'main', url: 'https://github.com/Haider7214/SpringBoot.git'

}

stage('Maven Build') {

def mvnHome=tool name:'maven-3.9.10', type:'maven';

mvnPath="${mvnHome}/bin/mvn";

sh "${mvnPath} clean package";

echo 'maven build success';

}

}

Multi Branch Pipeline in Jenkins

In Git repo, there will be multiple branches -> main branch, master, developer, release, feature

During git cloning, we got to specify which branch we are cloning from. Same Jenkins pipeline cannot work for all branches then if we are specifying the branch name explicitly

--> Creating different Jenkins pipeline for every branch would be difficult

--> We can create one common pipeline and build the code from multiple branches at a time with the help of “Multi-branch pipeline” concept

--> When we create Multi-branch pipeline, it will scan all the branches in given Github repo and execute pipeline for all the branches

Whichever branch only has the code change, only that branch will be built

Note: Whenever we run Multi-branch pipeline for second time, it will execute the pipeline for which code changes/commits have been done.