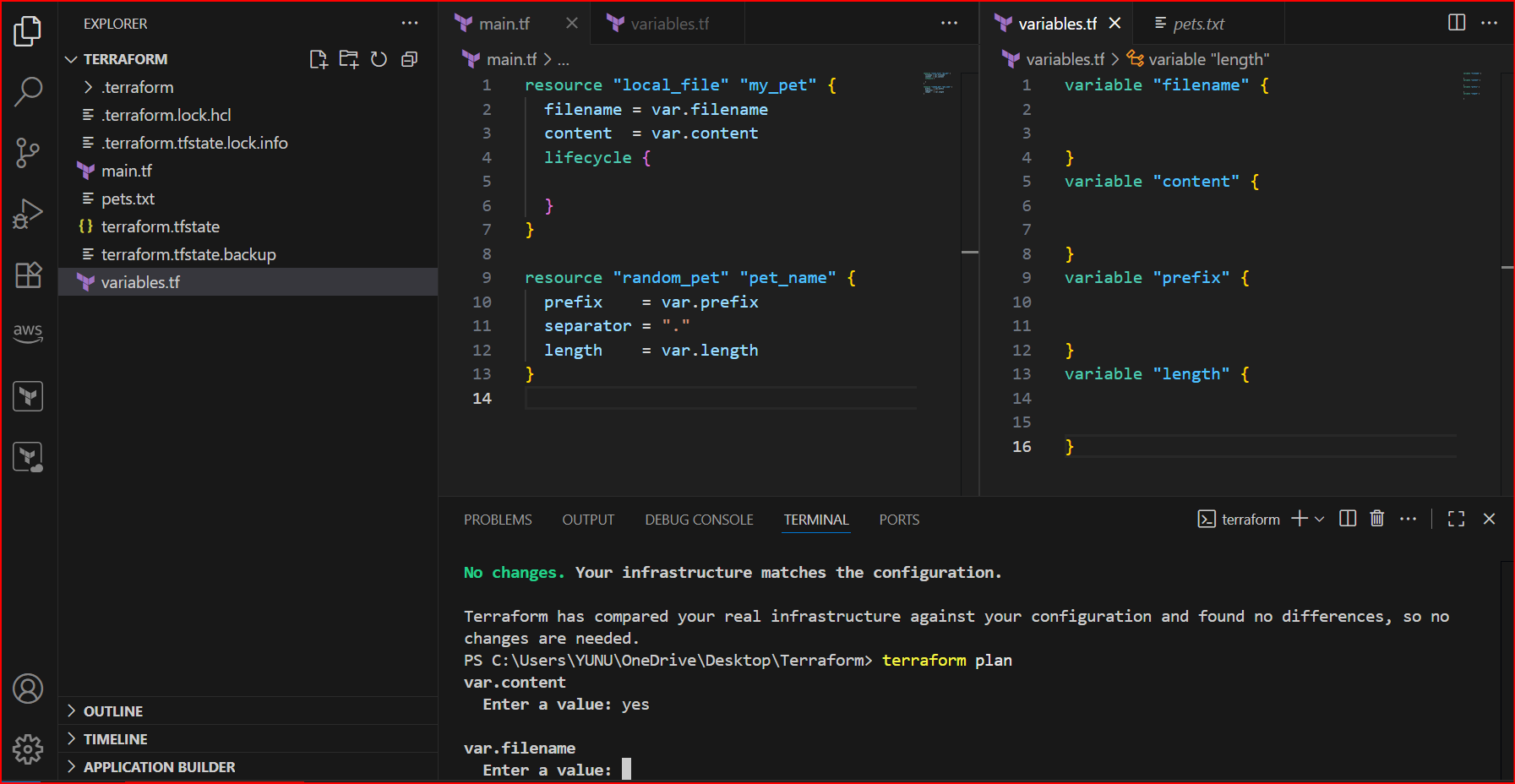
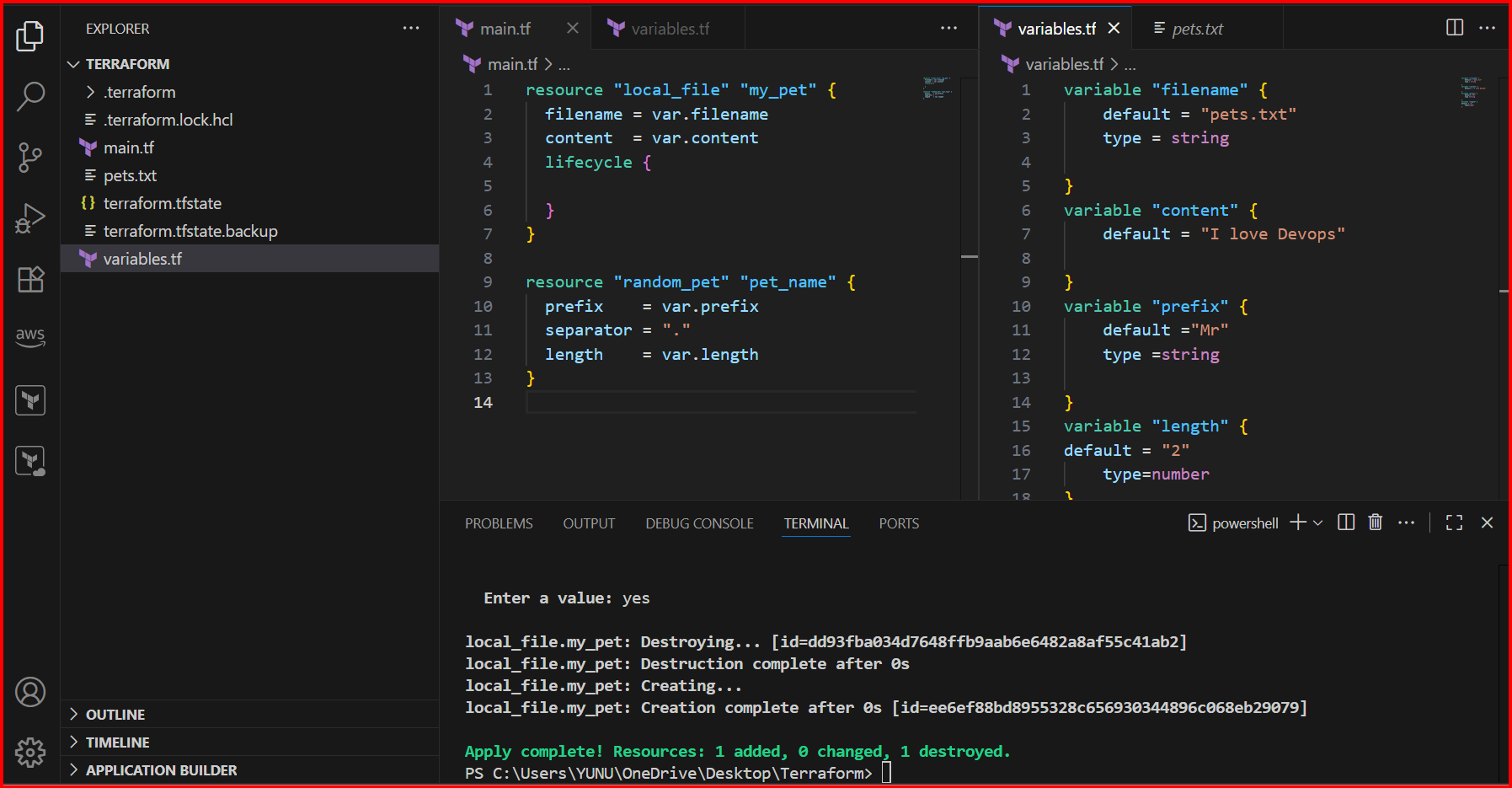
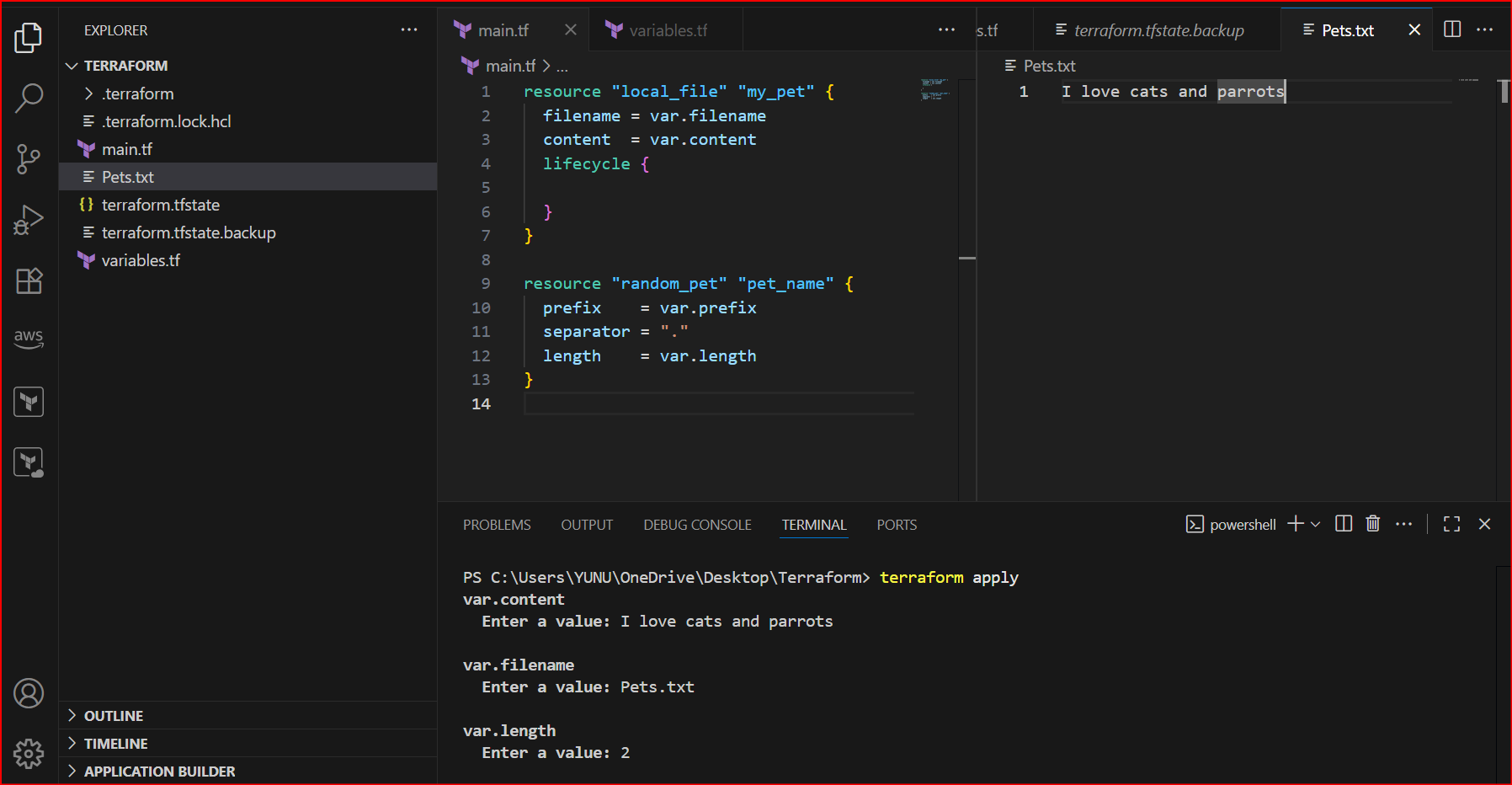
1. **Watch the Terraform-03 video.**
2. Completed the video
3. **Execute the script shown in the video.**
4. Here we have created variable file as emply as just created the sample of empty like variable has no default values

****

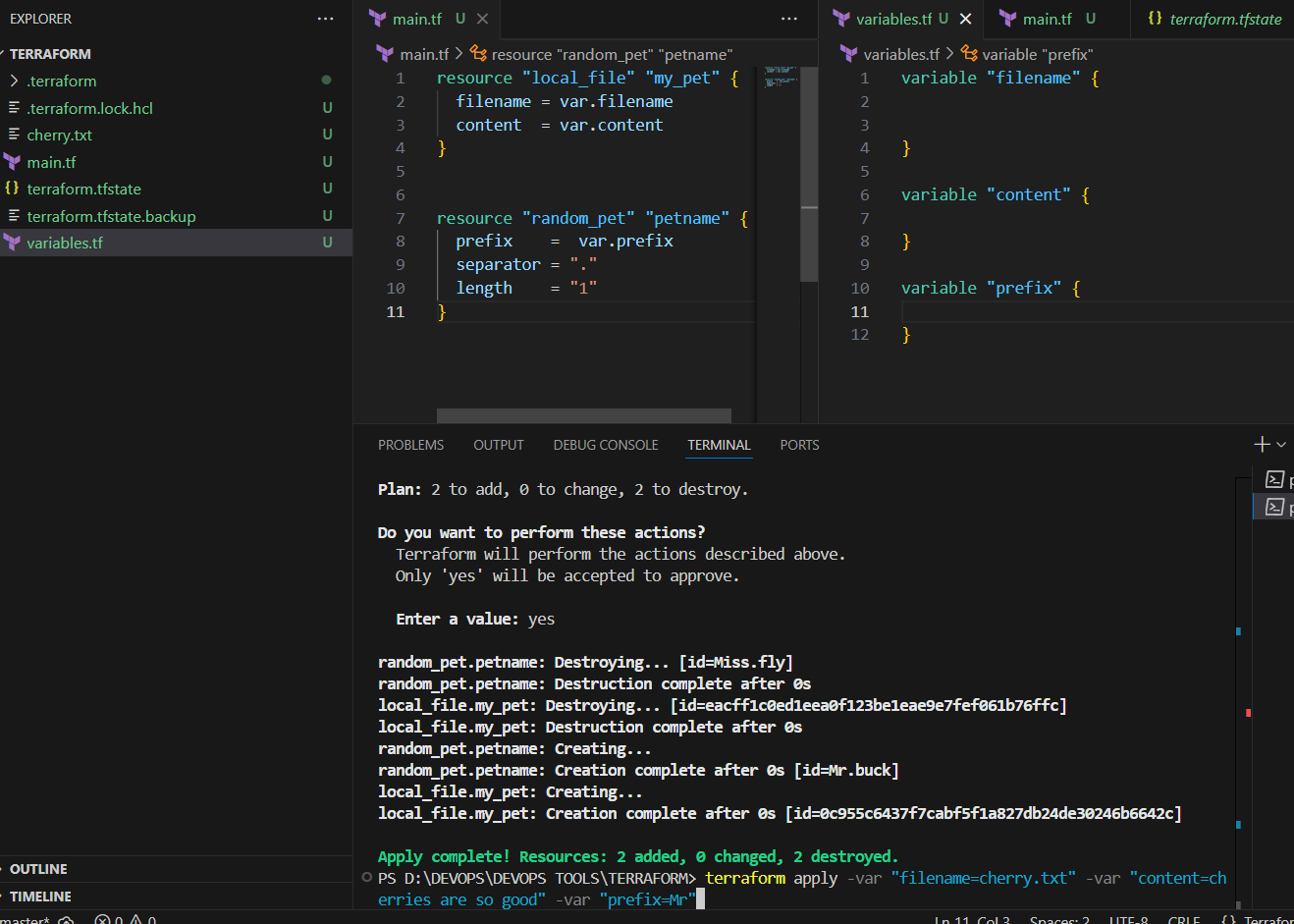
1. we have updated the variable file with 4 things filename, content, prefix, length in diff variables sections and than we have used command Terraform apply

****

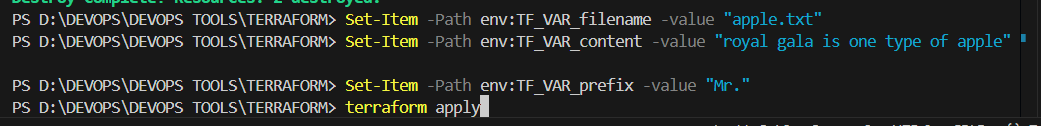
1. Than the pets.txt file will ask for enter the value after giving the command the terraform apply

****

1. Than the variable file with no default values we need to give command in command line flag using command --> terraform apply -var "filename=cherry.txt" -var "content=cherries are so good" -var "prefix=Mr"

****

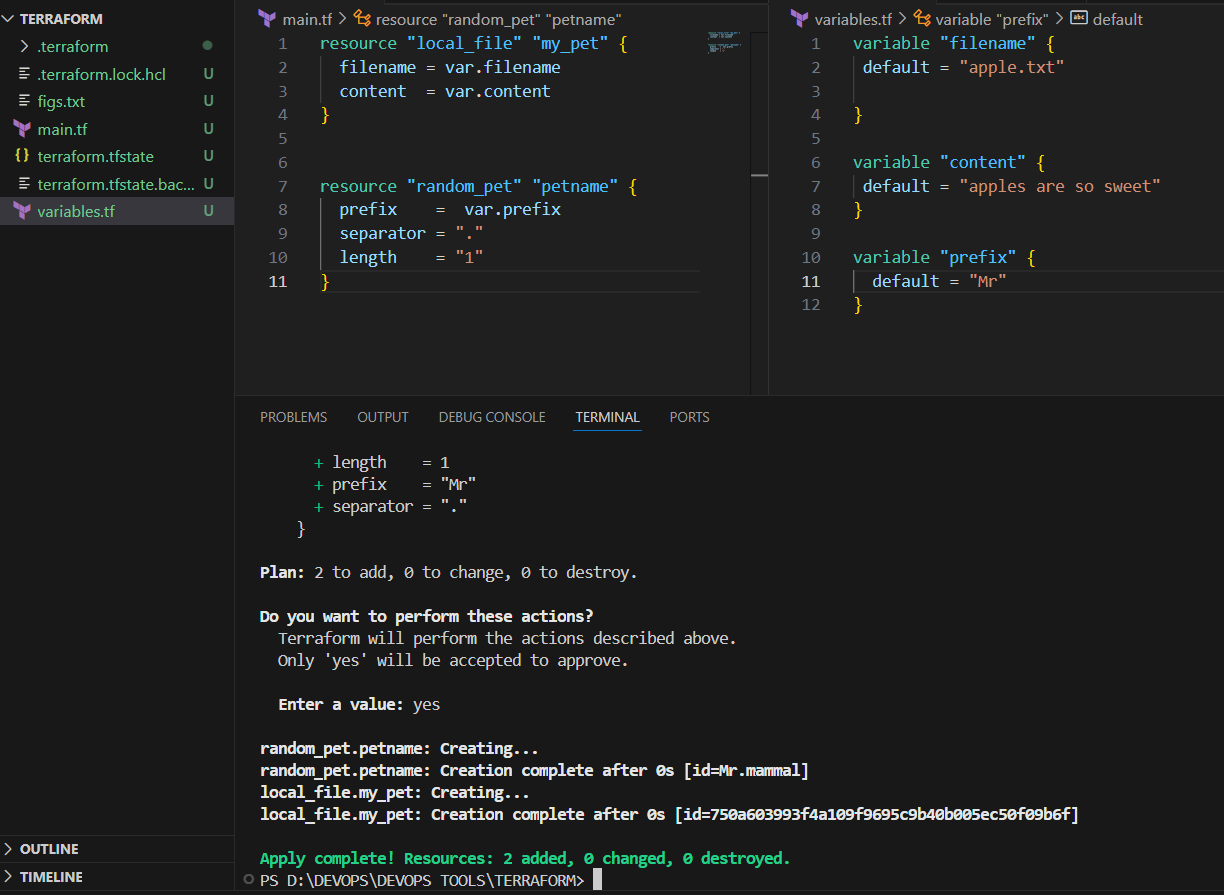
1. Now giving the command for environment variables

****

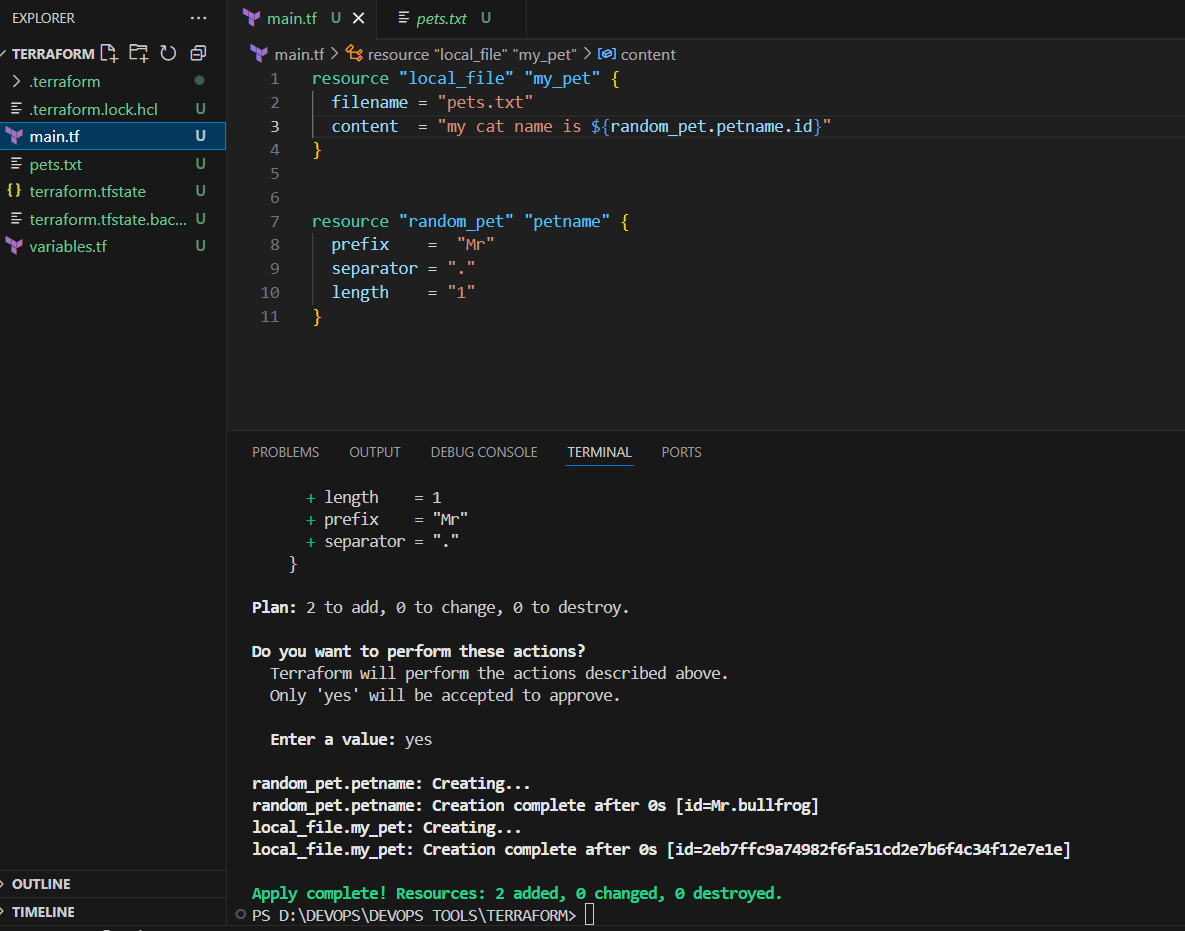
1. Set-Item -Path env:TF\_VAR\_filename -value "apple.txt"

Set-Item -Path env:TF\_VAR\_prefix -value "Mr."

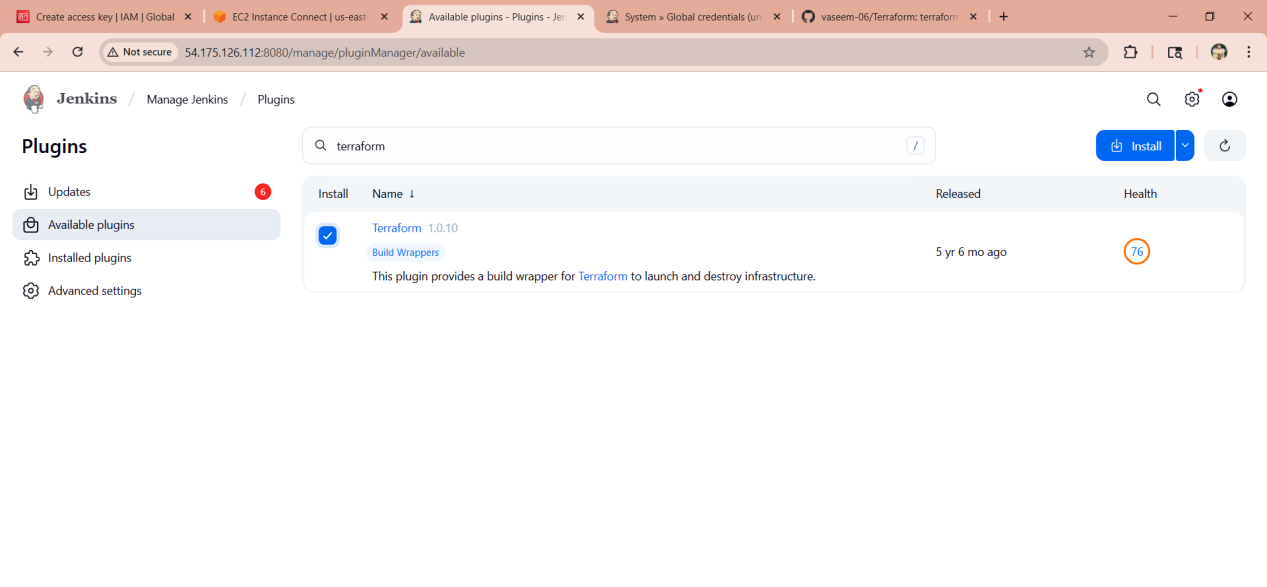
1. Than give command terraform apply
2. Now here it added 2 resources and 0 destroyed and 0 changes

****

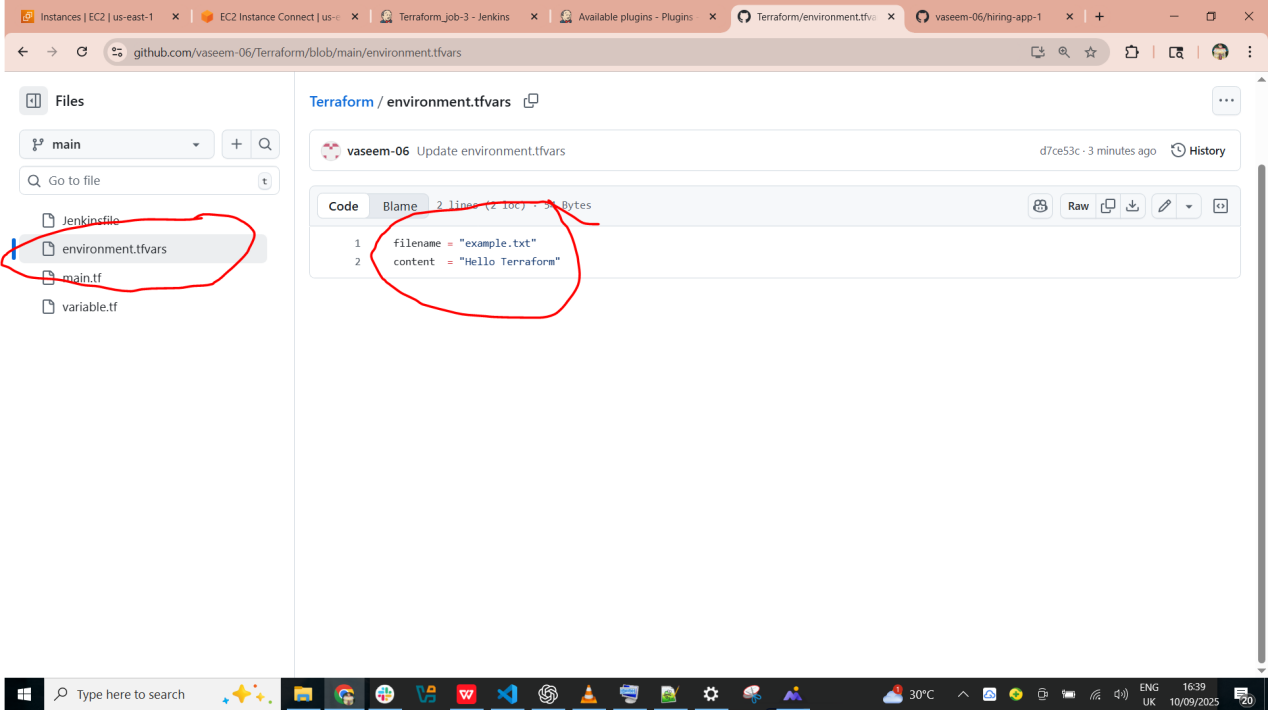
1. In resource attribute variable the main.tf in random provider it will give petname prefix separator and length terraform apply

****

1. **Integrate Terraform in Jenkins using the Terraform plugin.**
2. We have to install the Terraform Plugins in Plugins options



1. We need to create environment.tfvar file in github of terraform and it that I have gave filename as example.txt



1. Now in jenkins file need tp mention the script the with all the details belwo script

pipeline {

agent any

environment {

TFVARS = 'environment.tfvars' // change only here if you rename file

}

stages {

stage('Checkout Code') {

steps {

git branch: 'main', url: 'https://github.com/vaseem-06/Terraform.git'

}

}

stage('Terraform Init') {

steps {

sh 'terraform init'

}

}

stage('Terraform Plan') {

steps {

sh 'terraform plan -var-file=environment.tfvars'

}

}

stage('Terraform Apply') {

steps {

script {

if (fileExists(env.TFVARS)) {

sh "terraform apply -auto-approve -var-file=${env.TFVARS}"

} else {

sh "terraform apply -auto-approve"

}

}

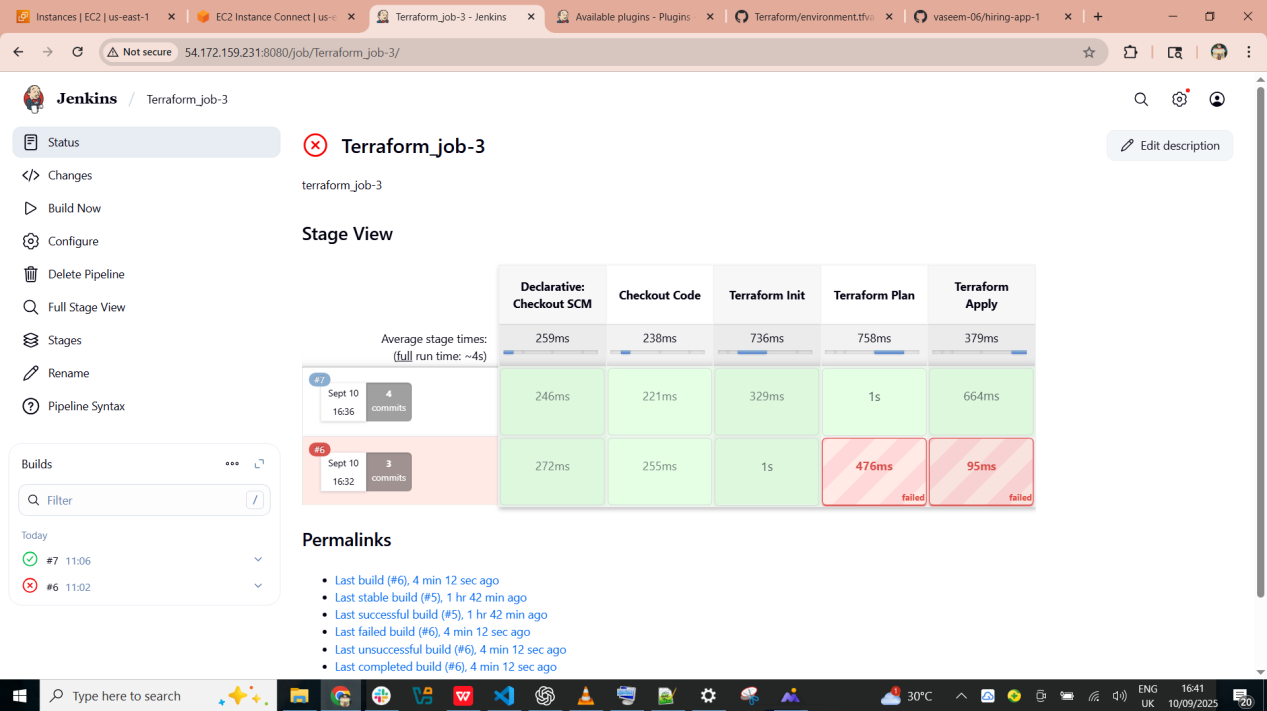
}

}

}

}

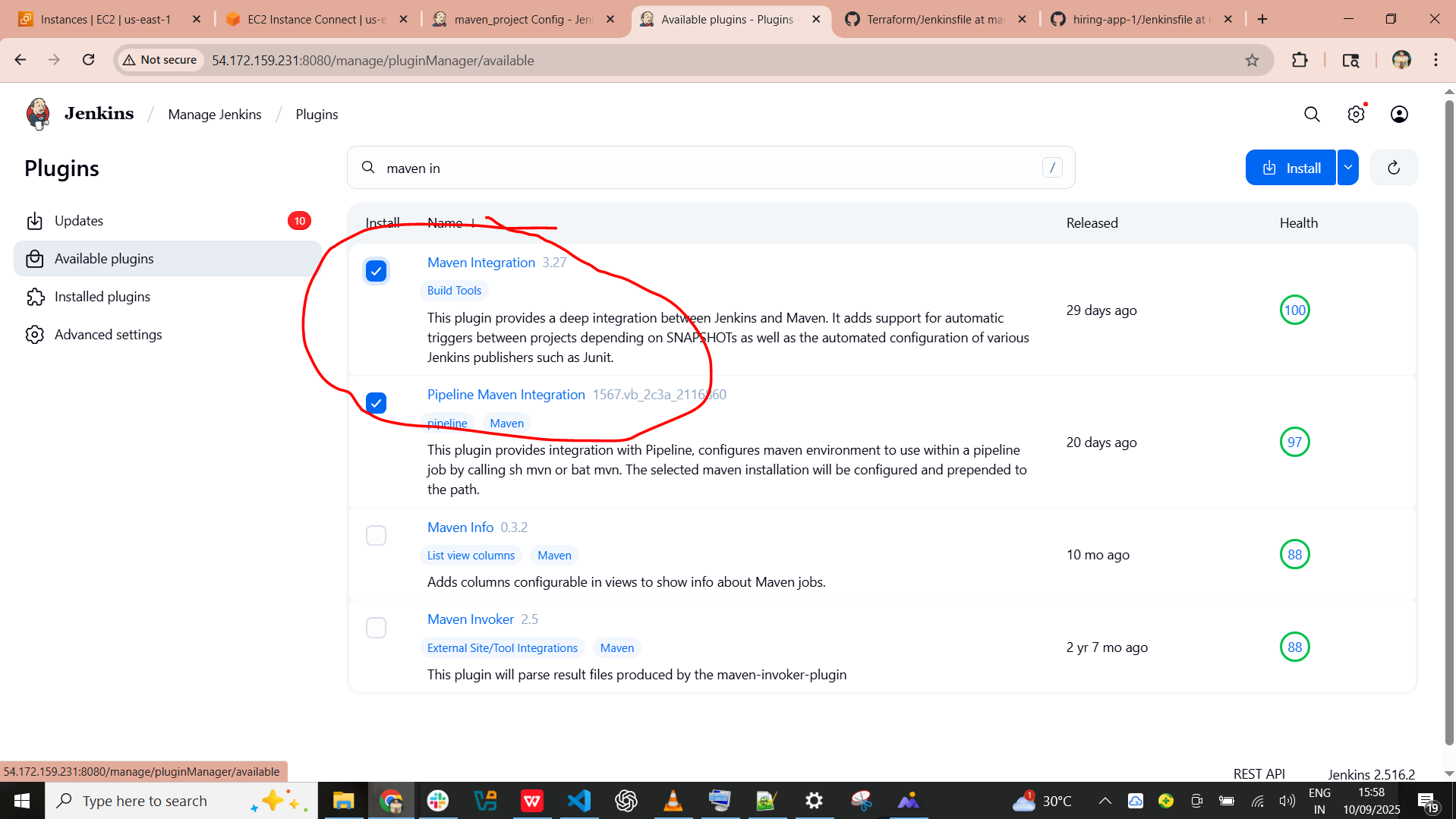
1. Now run the job it will get suucess with that all details

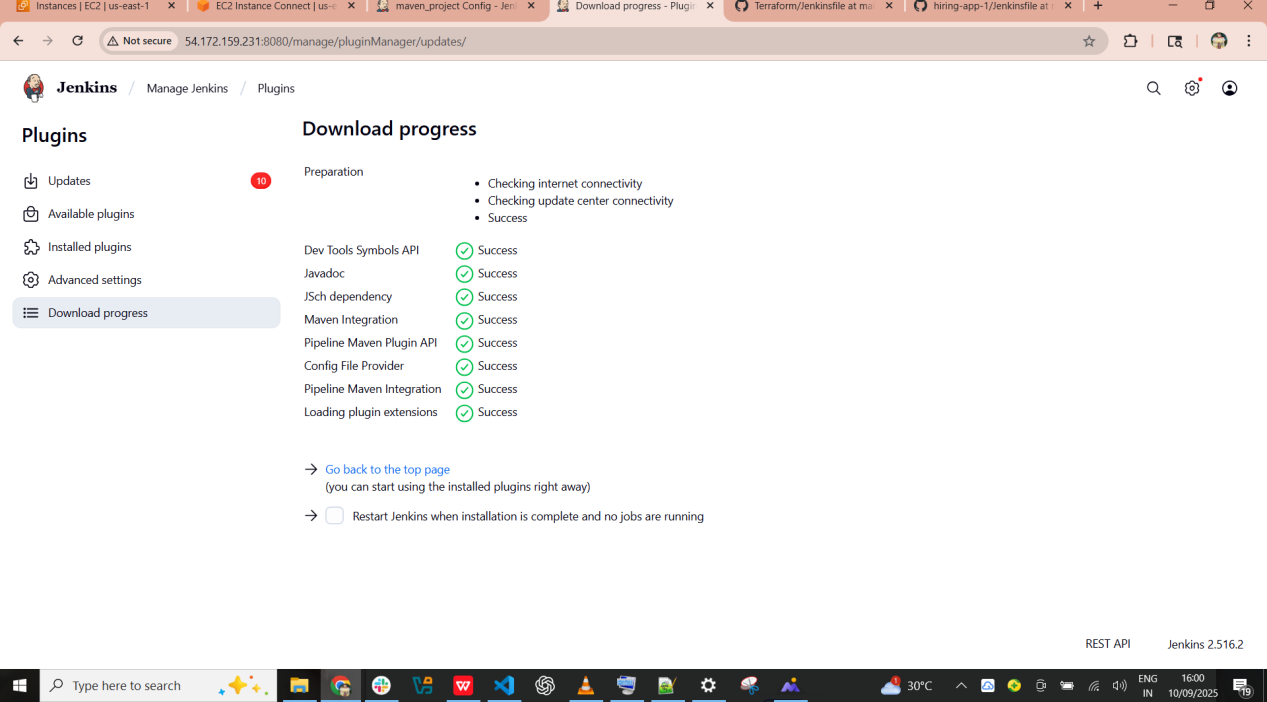


1. Now go to jenkins gitbash and go to **cd /var/lib/jenkins/workspace/ --> ls --> cd Terraform\_job-3/ --> ll ,** than will see that the environment.tfvar file will get trigger here

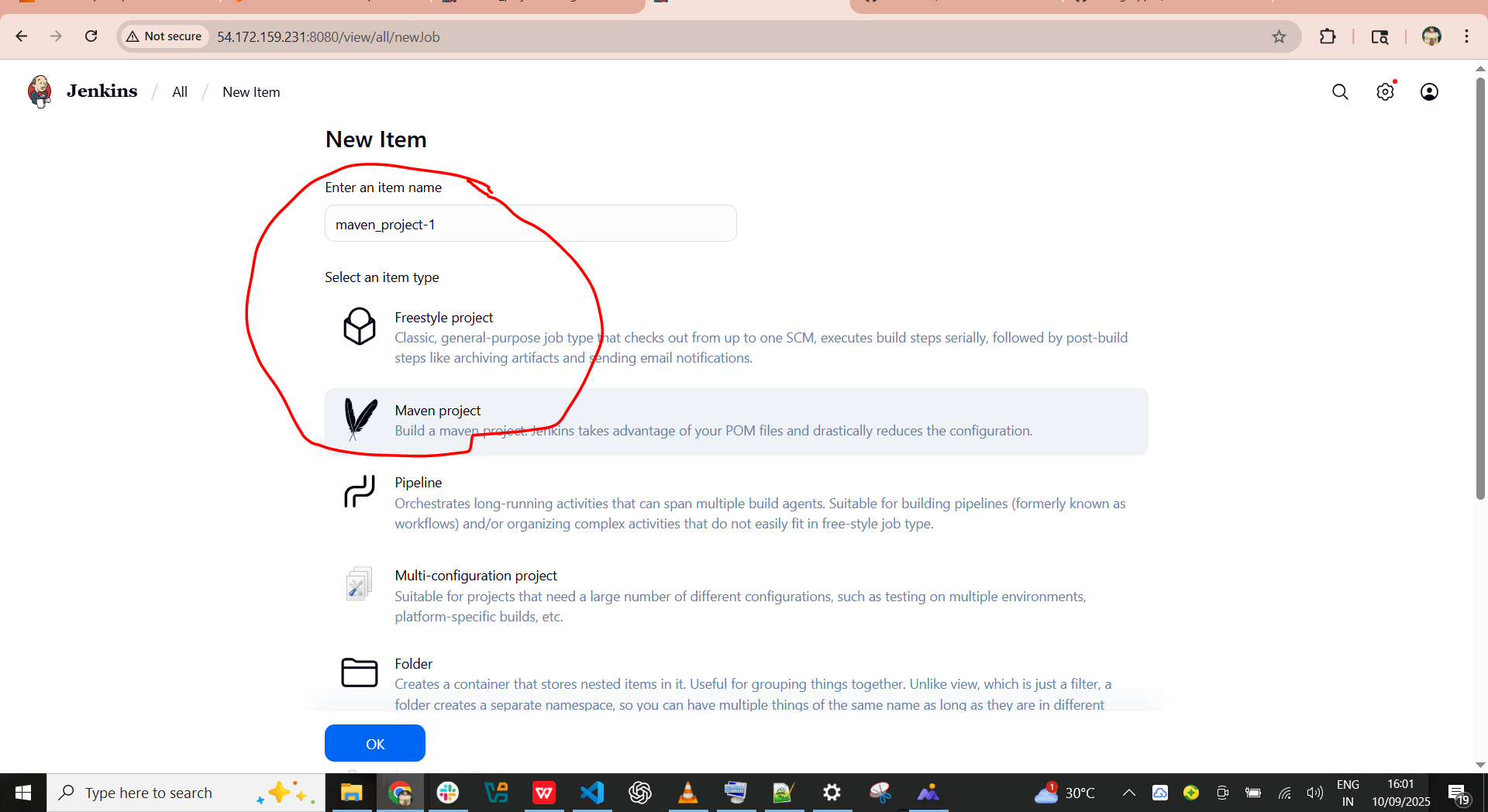


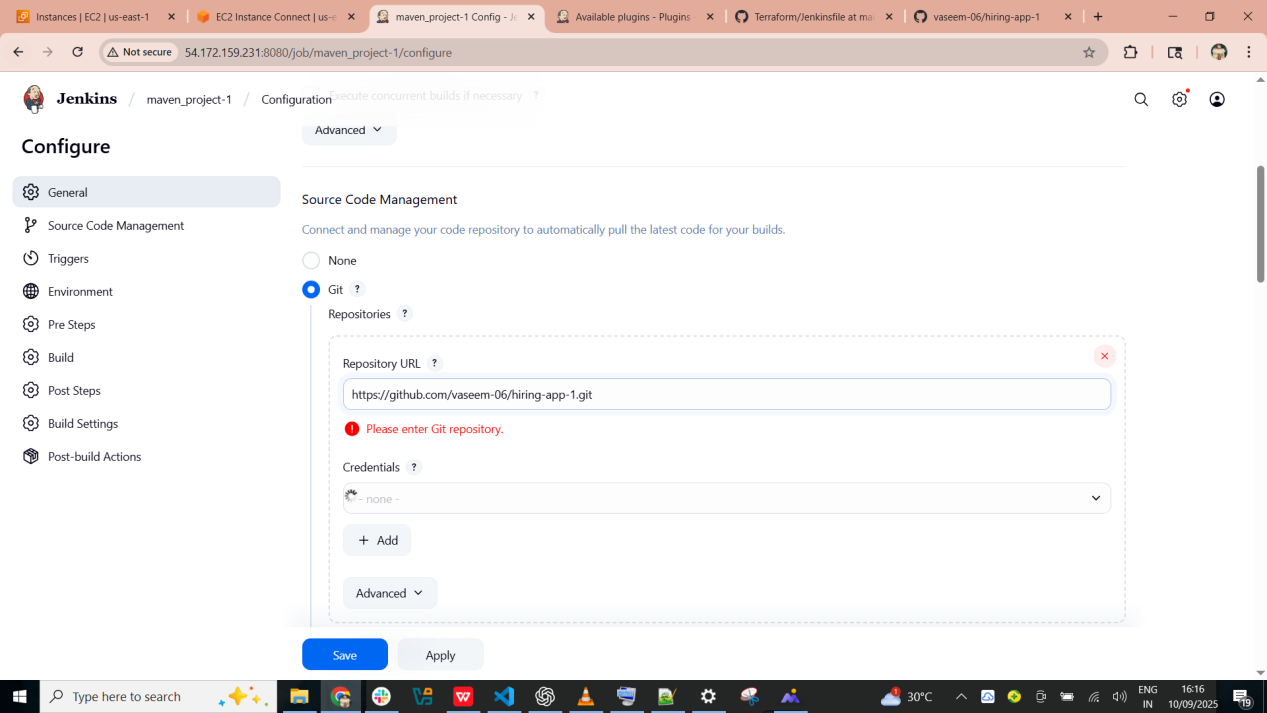
1. **Create one Jenkins job using Maven Project for the code below with two stages:**
   * **Stage 1: Git clone**
   * **Stage 2: Maven Compilation Code: [https://github.com/betawins/java-Working-app.git](https://github.com/betawins/java-Working-app.git" \t "https://app.slack.com/client/T03TRQ064Q0/_blank)**
2. Install maven plugins in manage jenkins



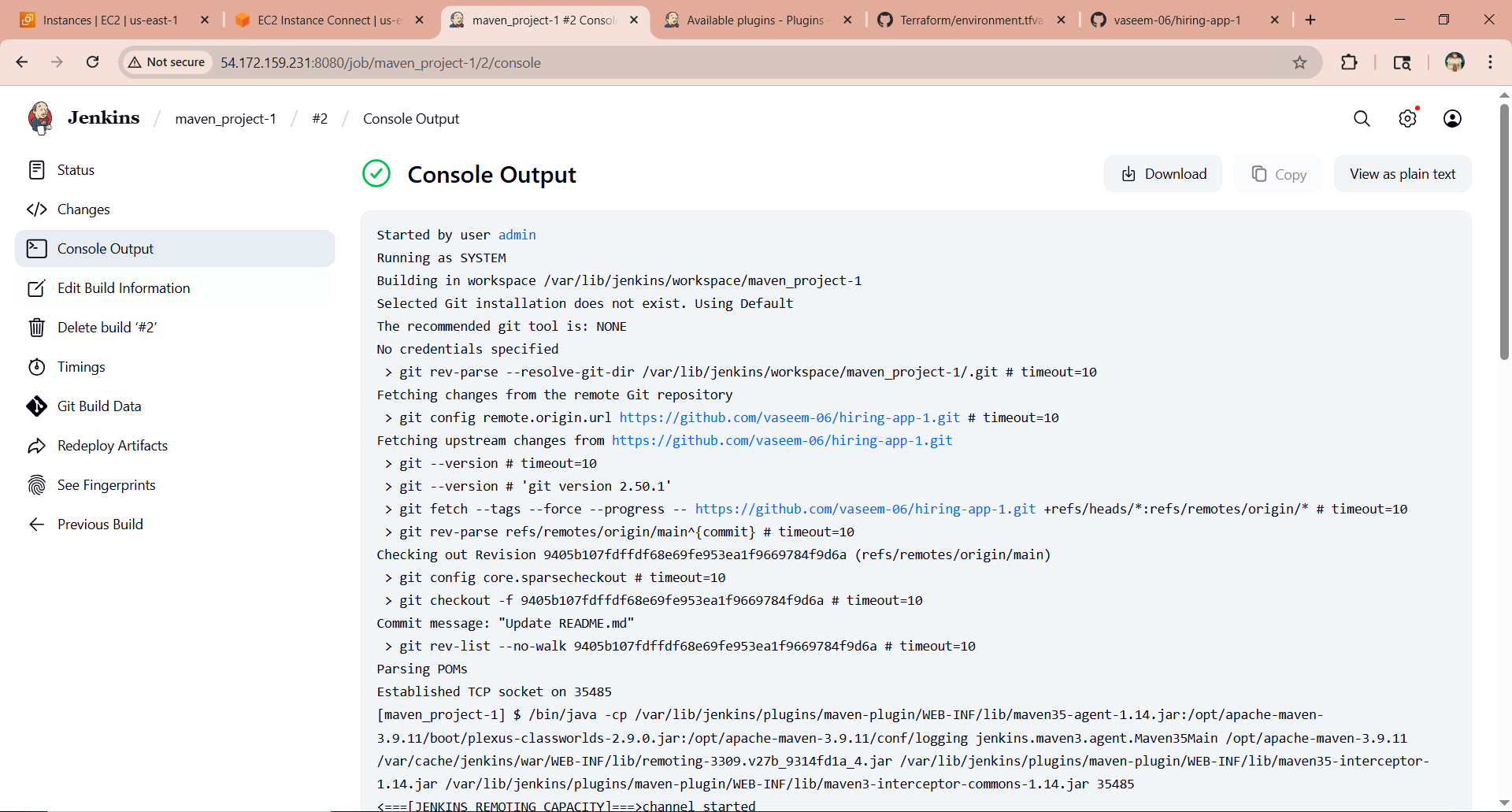


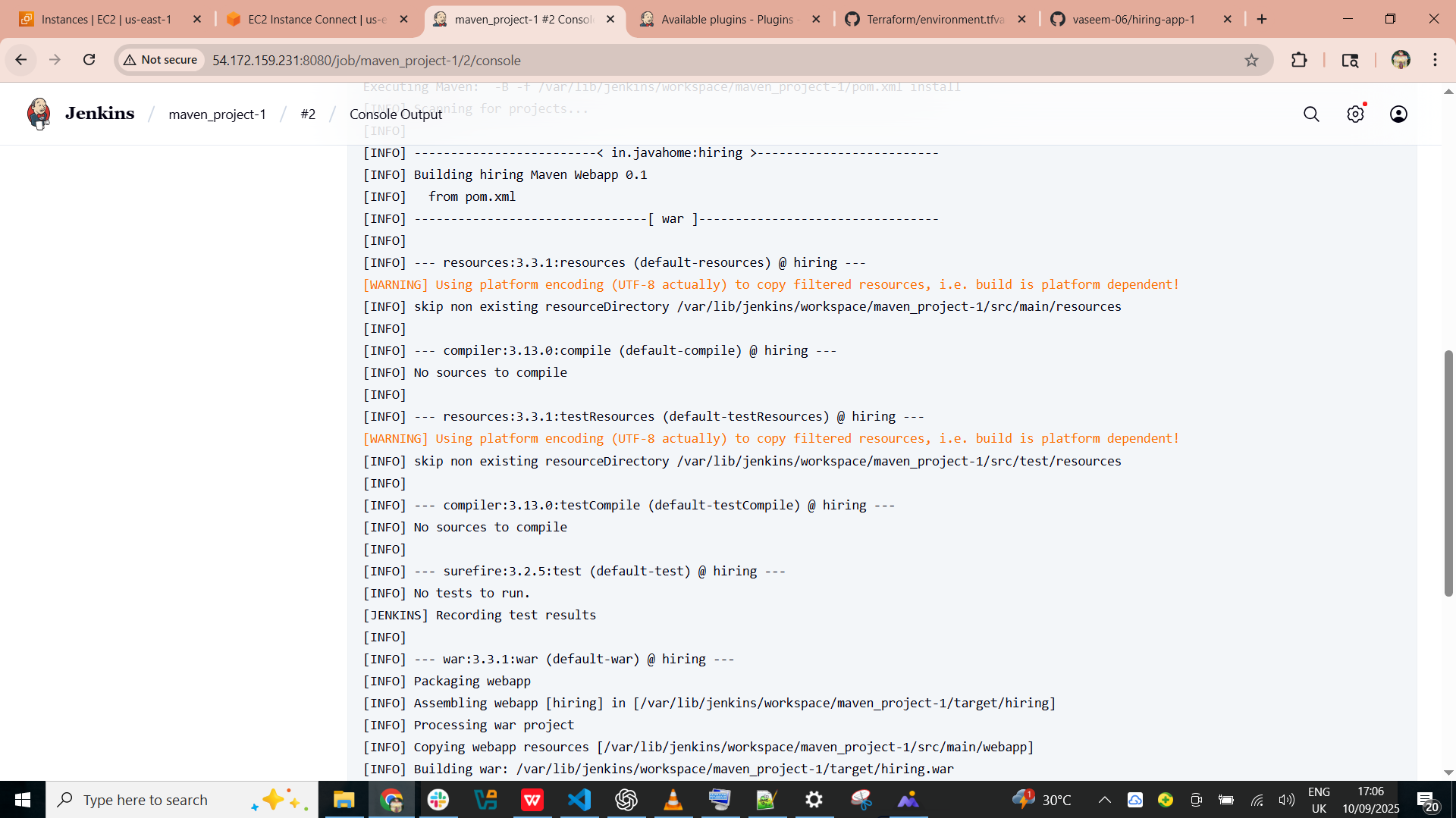
1. Now create job maven and selext mavn project



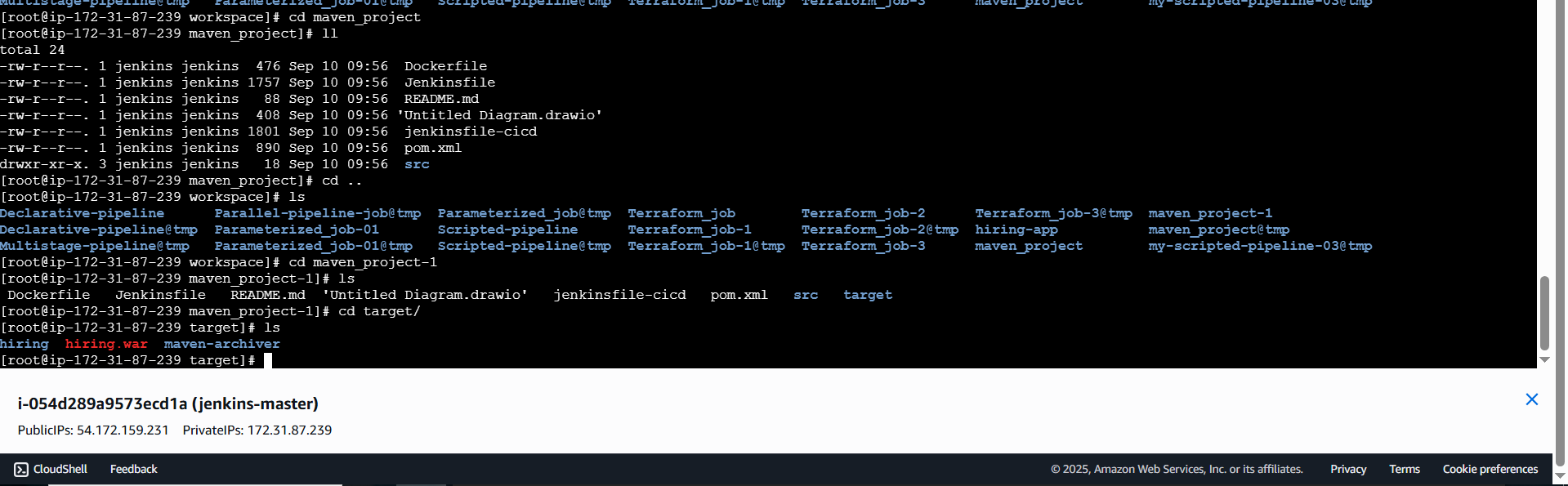


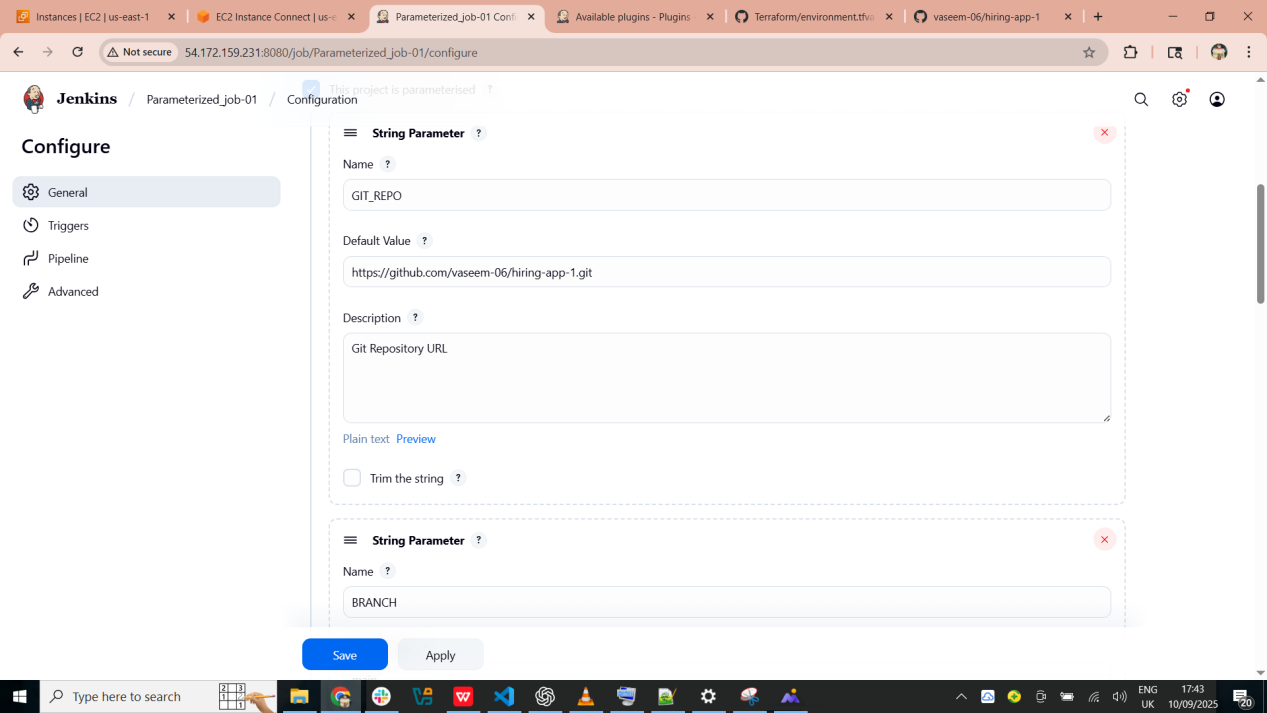
1. Here the job got success with git clone and maven compilation

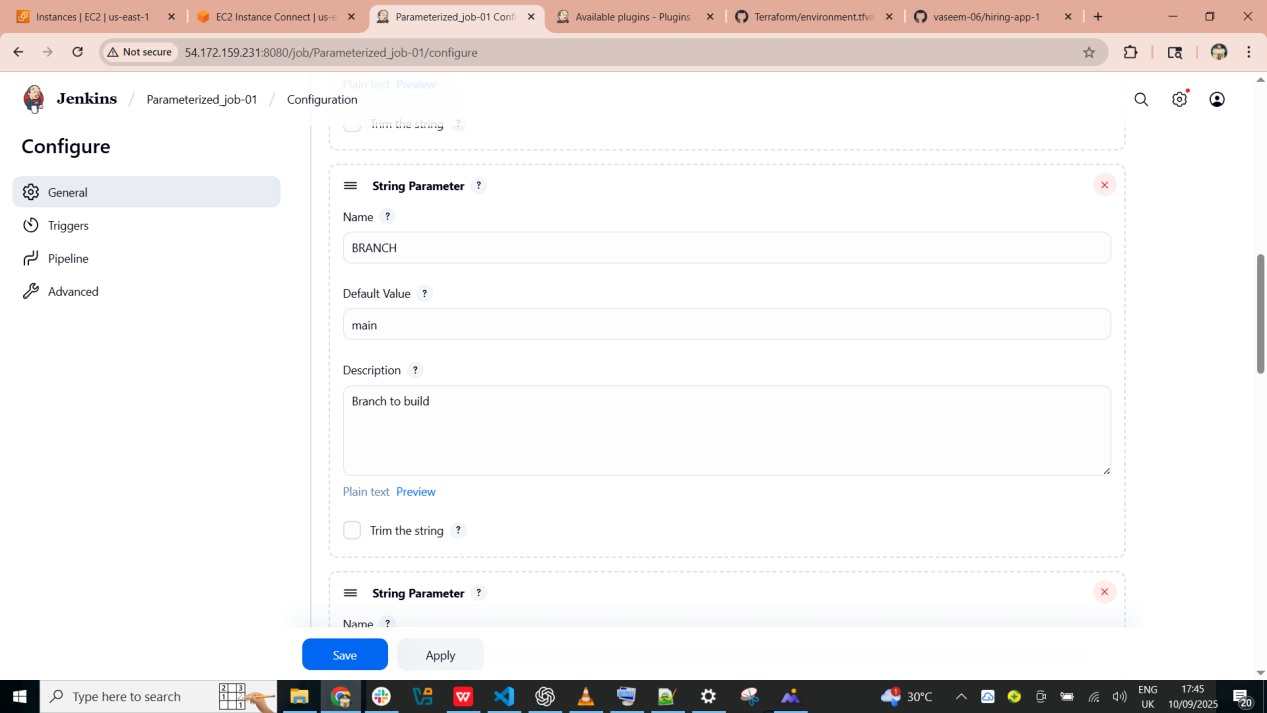


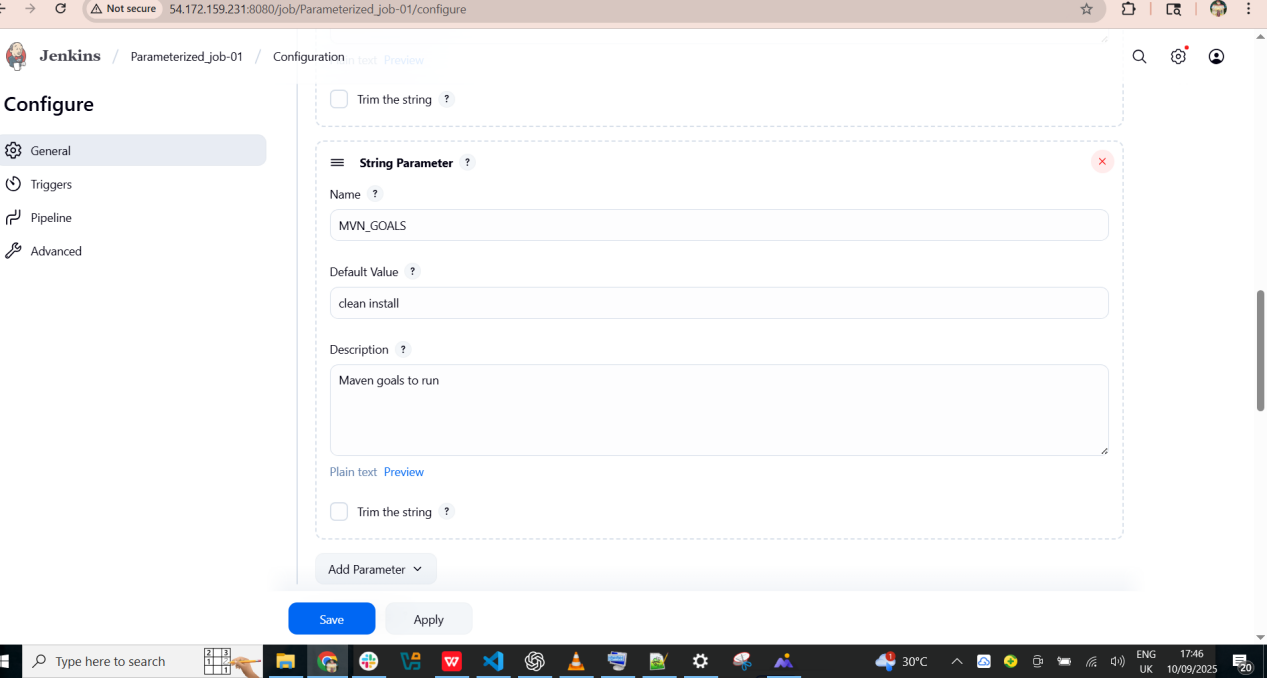


1. Now go to jenkins gitbash and check by command --> **cd /var/lib/jenkins/workspace/ --> ls --> cd maven\_project-1/ --> ll ,**



1. **Use the same code and create a parameterized job in Jenkins with:**
   * **Stage 1: Git clone**
   * **Stage 2: Maven Compilation Code: [https://github.com/betawins/java-Working-app.git](https://github.com/betawins/java-Working-app.git" \t "https://app.slack.com/client/T03TRQ064Q0/_blank)**
2. Here we have create job as parameterized\_job-01 
3. Now select the parametes as below shown





1. Here select the pipeline srcipt option and enter the script eith details

properties([

parameters([

string(name: 'GIT\_URL', defaultValue: 'https://github.com/vaseem-06/hiring-app.git', description: 'Git repository URL'),

string(name: 'BRANCH', defaultValue: 'main', description: 'Branch to build')

])

])

node {

// Use Maven tool configured in Jenkins (Name = MAVEN\_HOME)

def mvnHome = tool name: 'MVN\_HOME', type: 'maven'

stage('Git Clone') {

echo "Cloning from ${params.GIT\_URL} (branch: ${params.BRANCH})"

git url: params.GIT\_URL, branch: params.BRANCH

}

stage('Maven Package') {

echo "Running Maven package with ${mvnHome}"

sh "${mvnHome}/bin/mvn clean package"

}

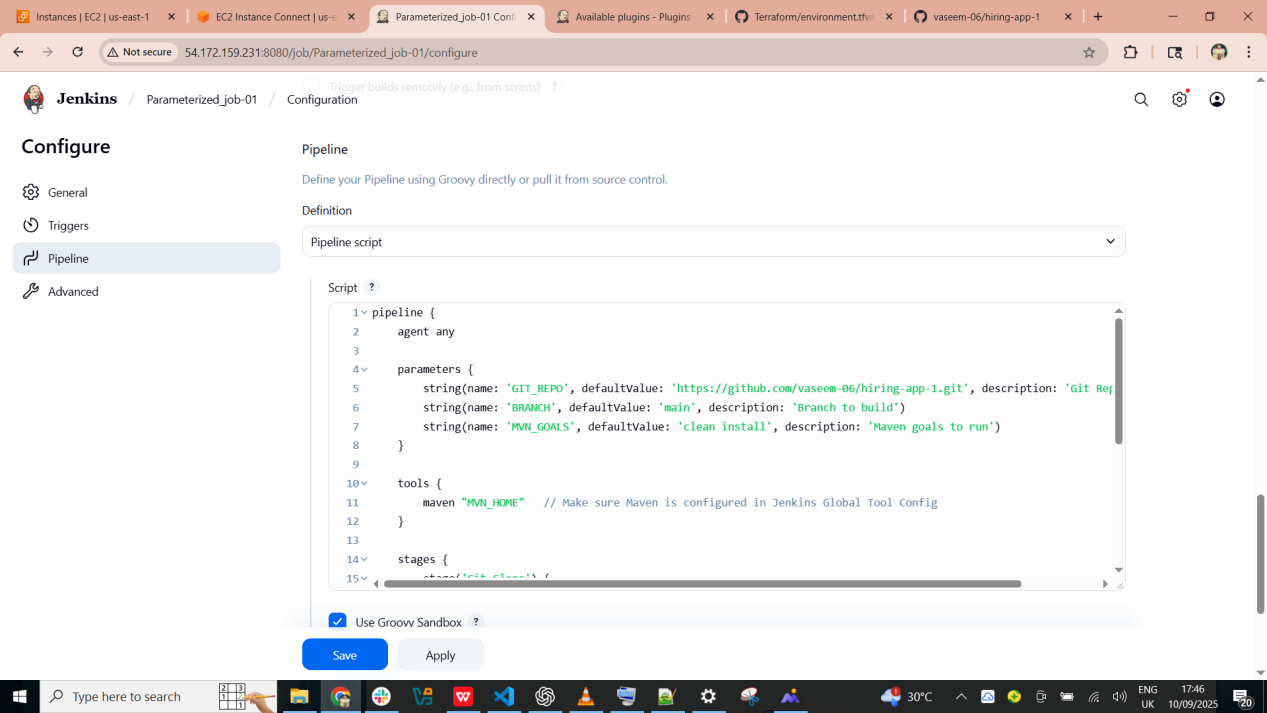
stage('Archive Artifacts') {

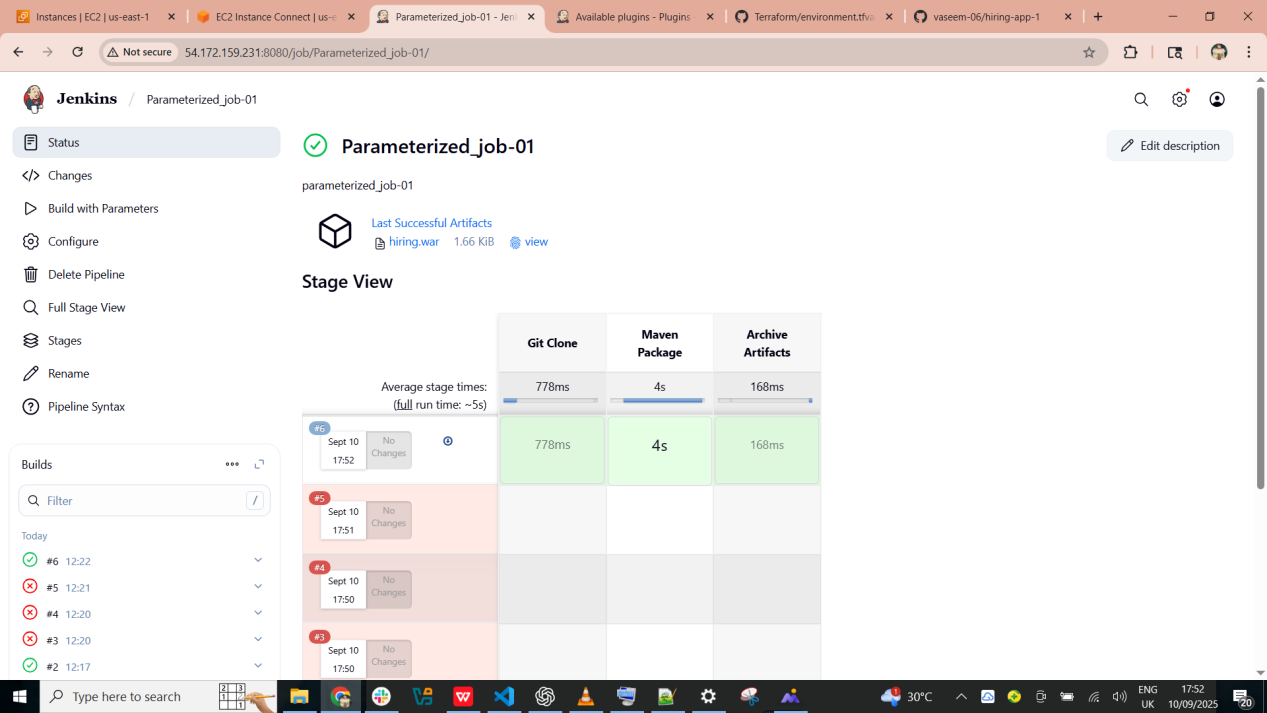
echo "Archiving JAR/WAR files from target/"

archiveArtifacts artifacts: 'target/\*.jar, target/\*.war', fingerprint: true

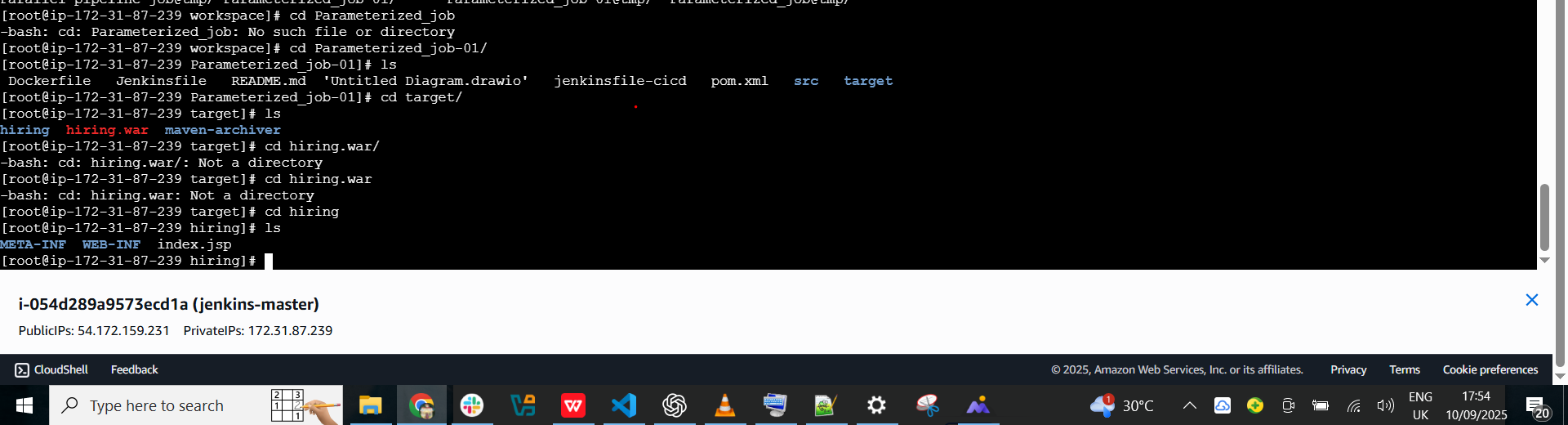
}

}





1. Now go to jenkins gitbash and use command **cd /var/lib/jenkins/workspace/ --> ls --> cd Parameterized\_job-01/ --> ll --> cd target/ --> ls**



1. **What are the global variables in Jenkins?**

In **Jenkins**, **Global Variables** are the predefined variables that Jenkins makes available automatically in every pipeline or freestyle job. **Global variables in Jenkins** are the built-in variables (like env, params, currentBuild, and system environment variables) that Jenkins provides automatically to every job. They let you easily fetch job details, control pipeline flow, and configure environments without extra setup.There are 3 types of global variable in jenkins

1. Environment Variables : These are automatically available and give runtime information about the build, job, or system.
2. BUILD\_ID → The current build ID (e.g., 2025-09-10\_20-20-31)
3. BUILD\_NUMBER → The build number for the job (e.g., #15)
4. JOB\_NAME → Name of the Jenkins job
5. WORKSPACE → Path to the workspace directory
6. **Predefined Script Variables:** Available directly inside the Jenkins pipeline Groovy script
7. currentBuild → Provides details about the current build (status, result, display name, etc.)
8. env → Lets you access environment variables (e.g., env.BUILD\_NUMBER)
9. params → Used for parameterized builds to access user-specified inputs
10. Jenkins-Specific Variables
11. BRANCH\_NAME → Available in multibranch pipelines, gives the branch name being built
12. CHANGE\_ID → Pull request/change request ID
13. CHANGE\_AUTHOR → Author of the change
14. GIT\_COMMIT → Current Git commit being built
15. GIT\_BRANCH → The branch that was checked out
16. **. Custom Global Environment Variables :** Admins can define custom global environment variables for **all jobs**
17. Configurable via **Manage Jenkins → Configure System → Global properties → Environment variables**