Medicure

Healthcare Domain

This project is developing an application which interacts about healthcare parameters.

1. Git - For version control for tracking changes in the code files .
2. Jenkins - For continuous integration and continuous deployment .
3. Docker - For containerizing applications .
4. Ansible - Configuration management tools.
5. Selenium - For automating tests on the deployed web application.
6. Terraform - For creation of infrastructure.
7. Kubernetes – for running containerized application in managed cluster.

Here is the EC2 server created with following specifications to act as master server. It is configured with java, maven, docker, Jenkins, Ansible and Terraform on the instance, this server acts as master server, which is used to implement CI/CD pipeline script.

Public IP: *3.110.151.185*

IAM Role: srija with EC2 Full Access.

Java Version: jdk-17

Maven version: maven 3.6.3.

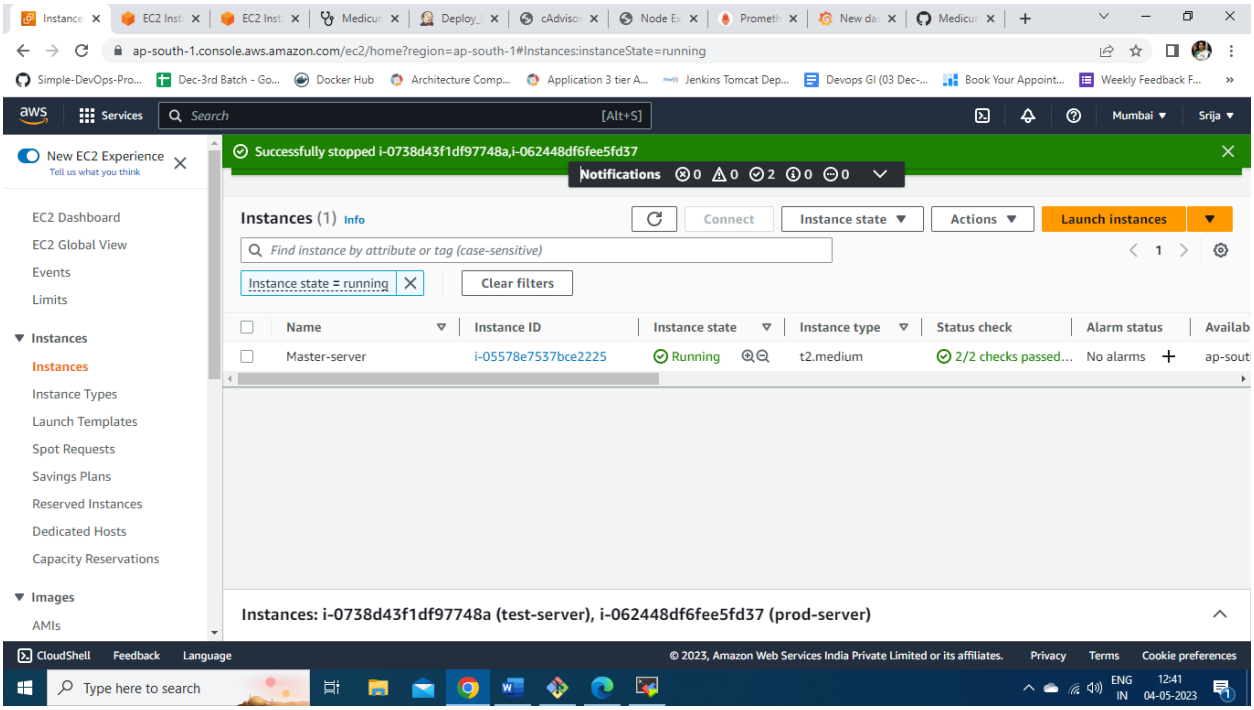
Ansible Version: 2.14.5

Terraform Version: v1.4.6

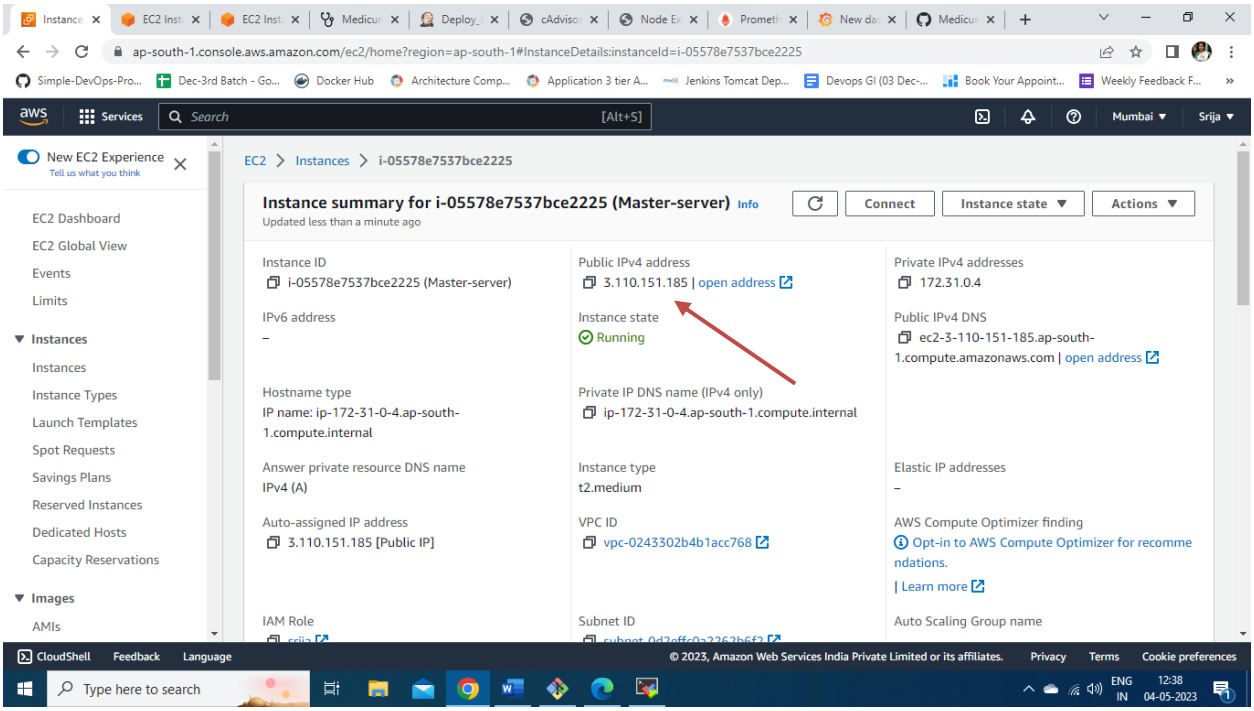
Docker Version: 20.10.21

Jenkins configured on URL: *3.110.151.185:8080*

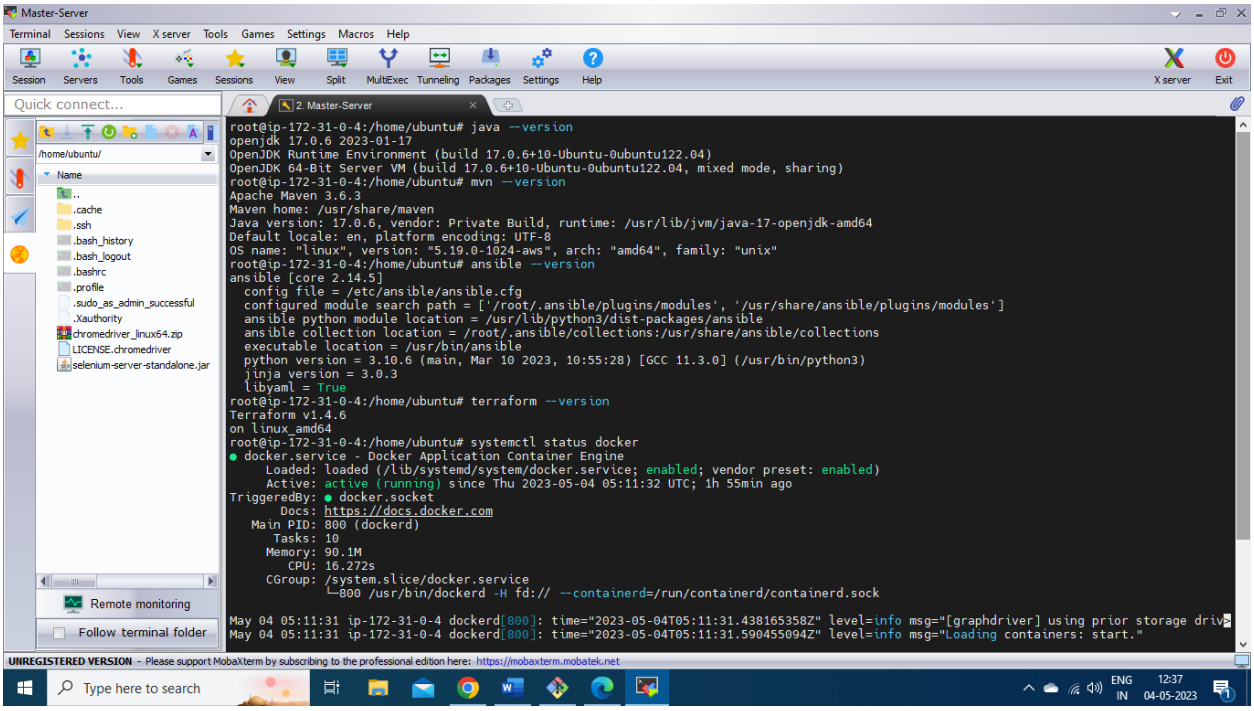
This is the EC2 dashboard showing that master-server is up and running.

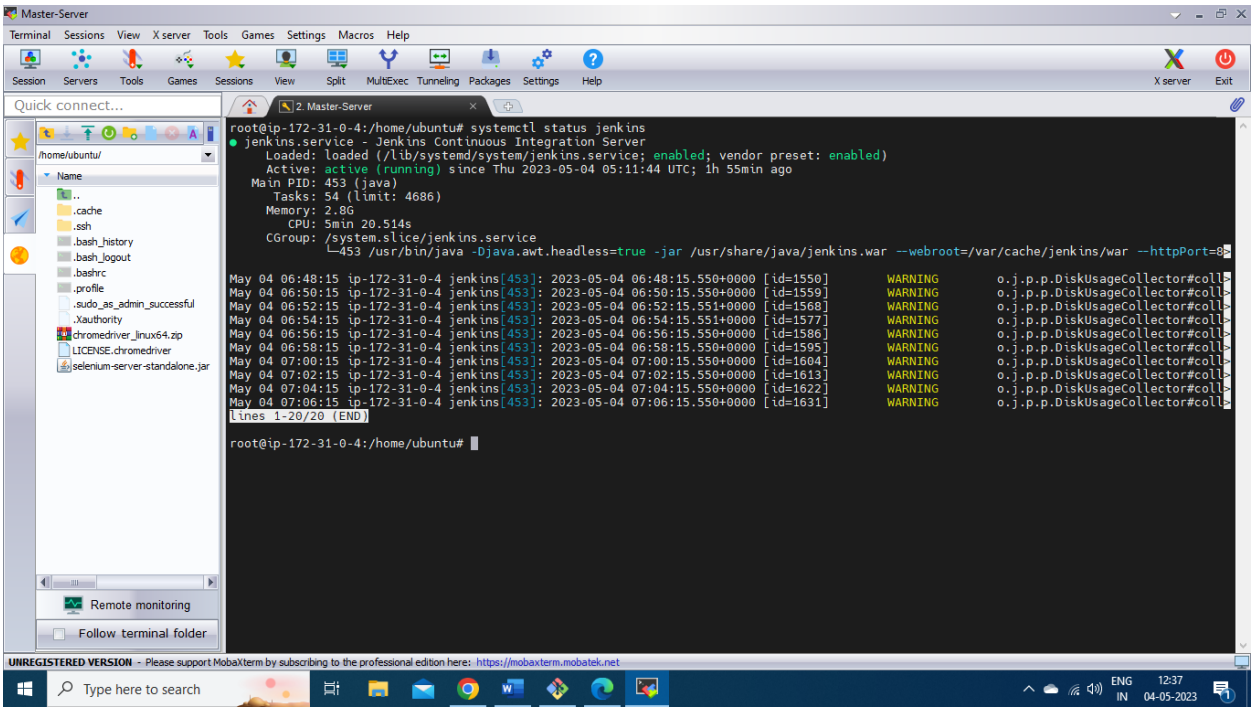


Here is the screenshot of IP address of master server configuration.

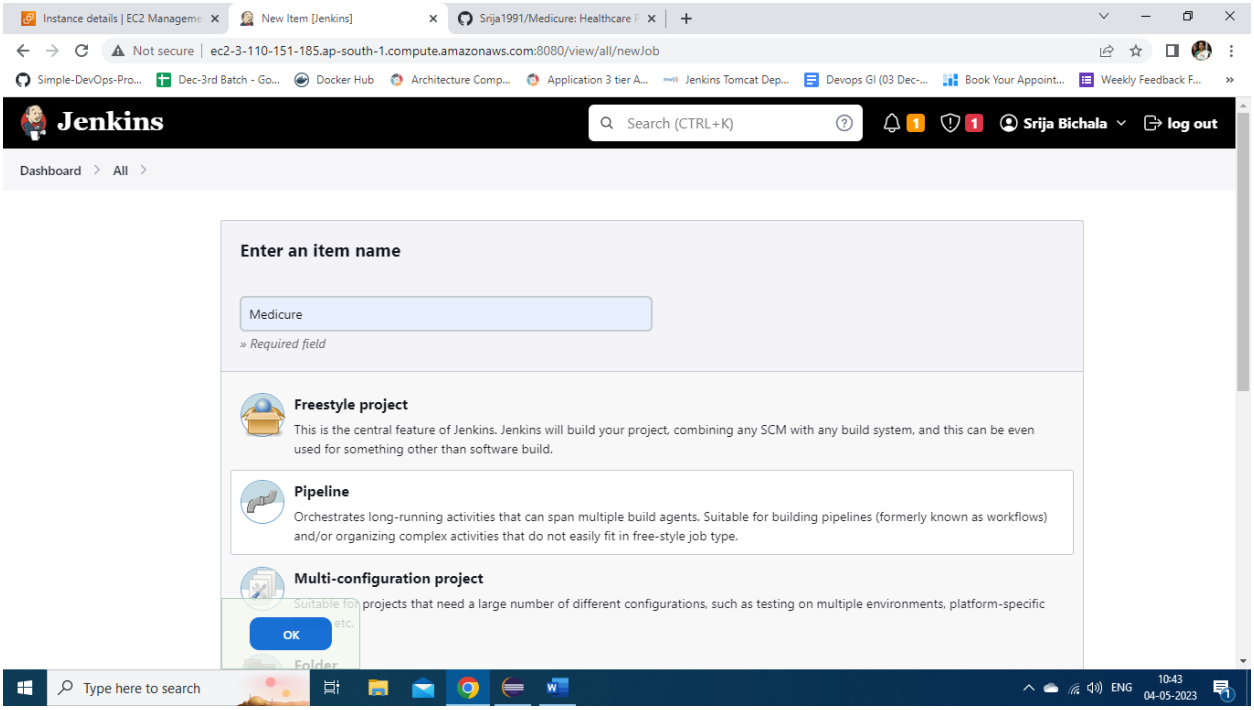


This is the figure showing master server is connected through SSH connection using mobaxterm and all the software packages installed with their versions and status.

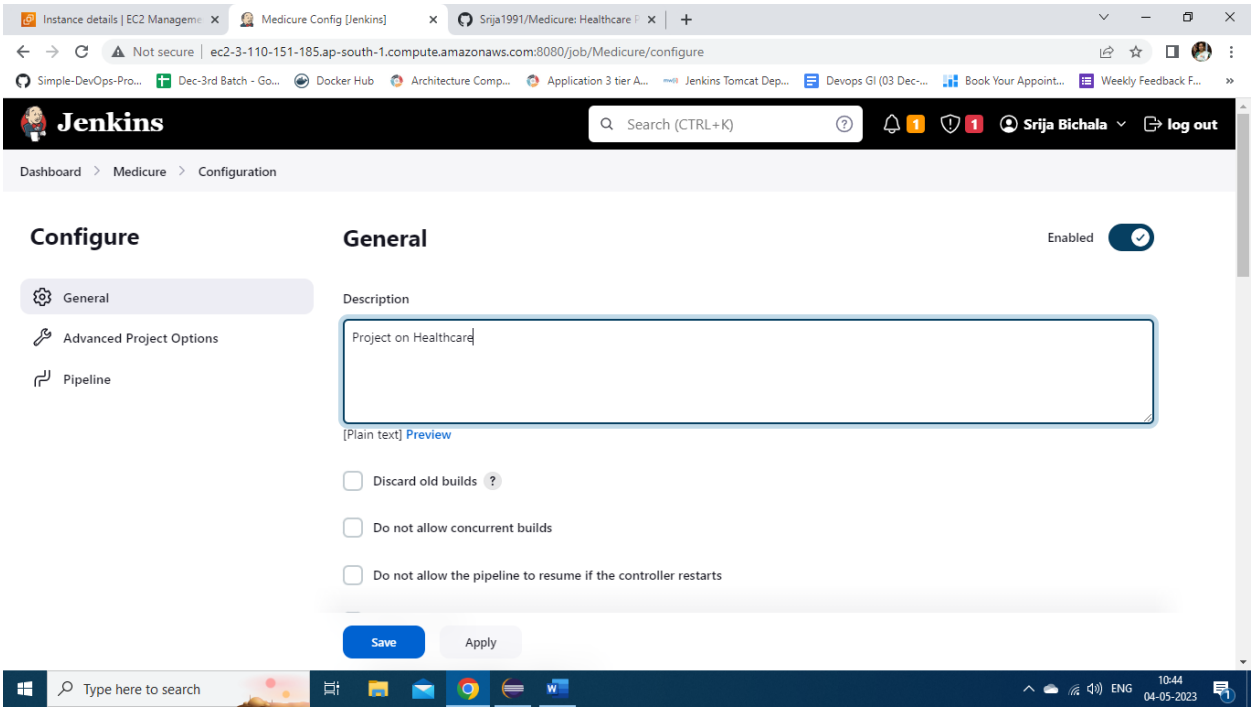




Jenkins is accessed on URL : 3.110.151.185:8080 port and a new job is created named ***Medicure*** to build the packages using maven build tool, then create docker image out of it using ***Docker file*** and then push the docker image onto Docker hub and use the image later to create a deployment on test server created using terraform and configured using ansible tool.

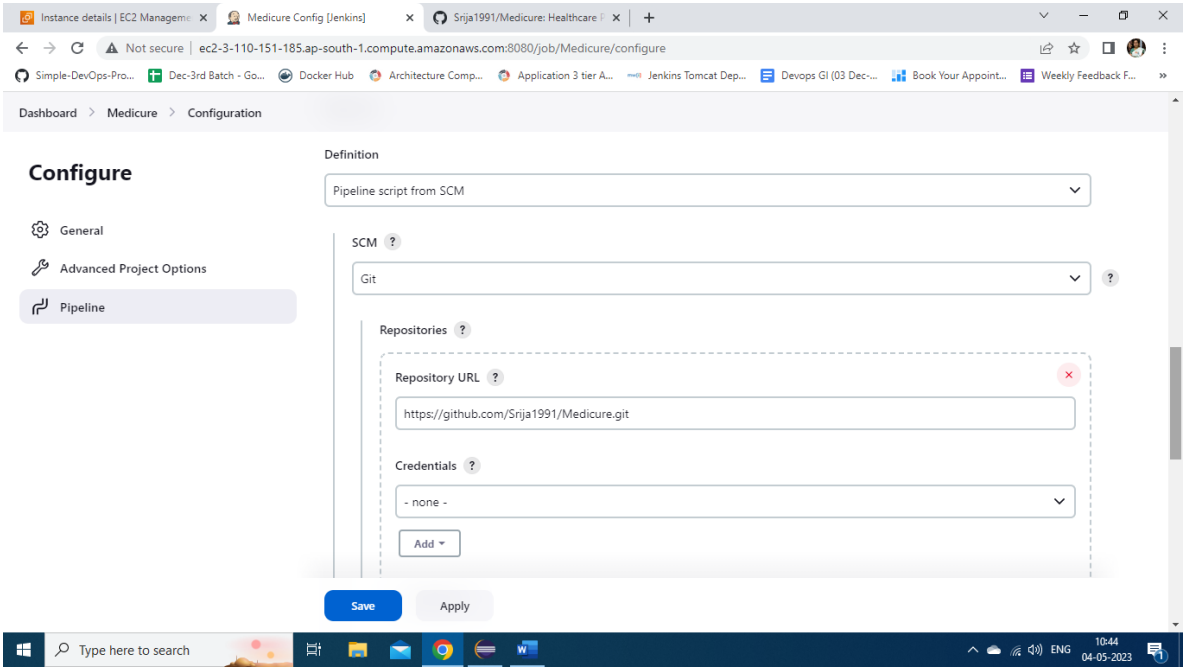


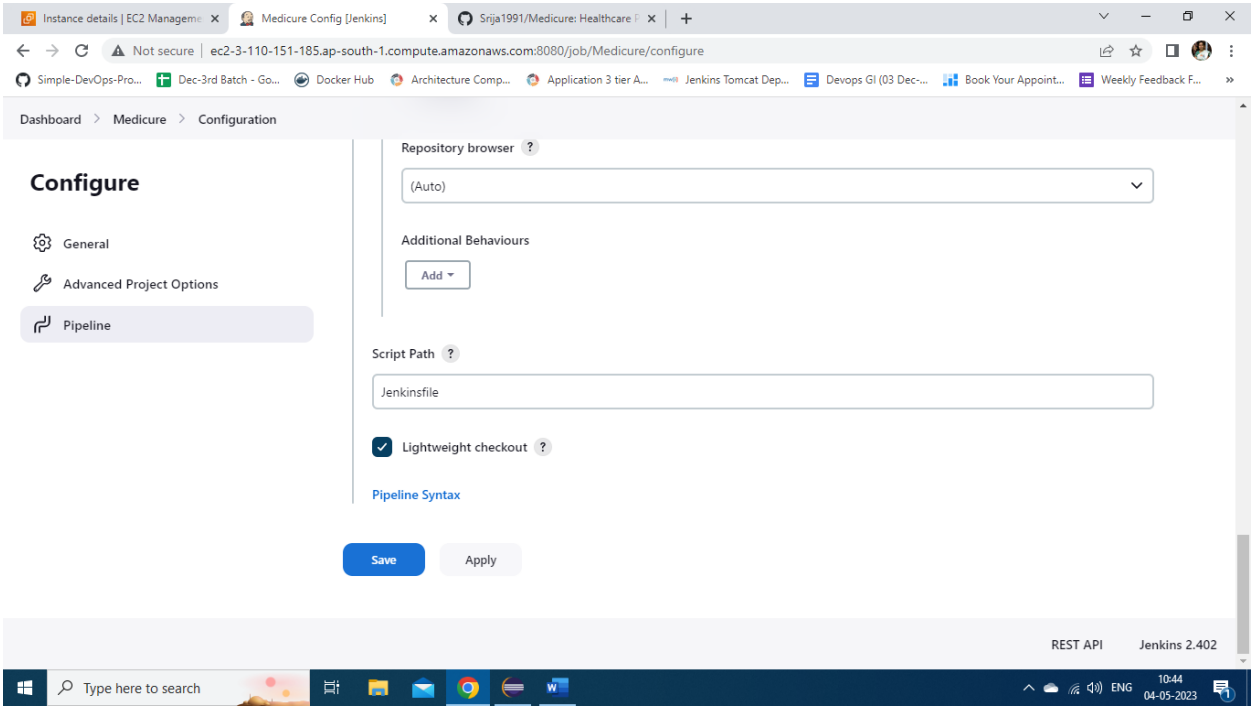
Here is the description of job Medicure.



The source code for this job is fetched through Git hub and the file name to execute pipeline script is "***Jenkinsfile***".

Git URL: https://github.com/Srija1991/Medicure.git.



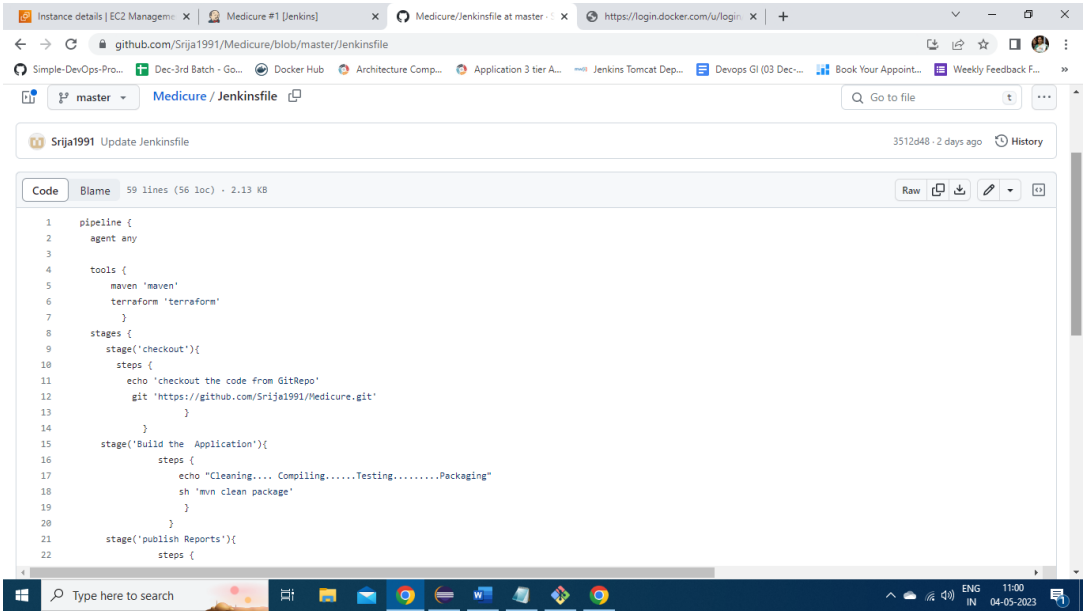


Here is Jenkins file illustrating the flow of CI/CD pipeline.

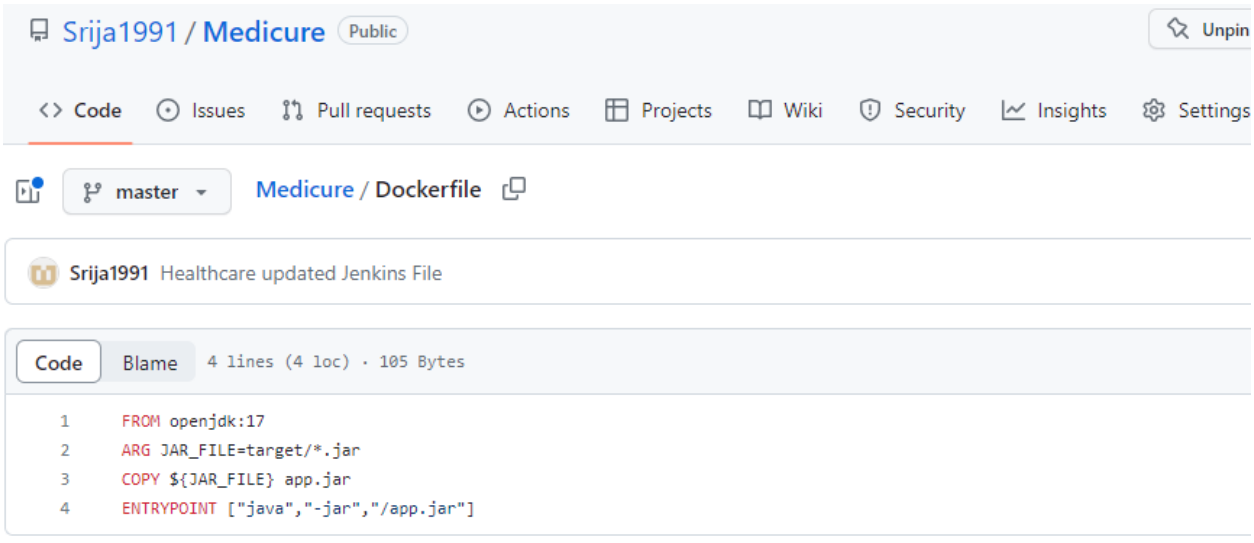
Stage 1: Git checkout : Use the code available on git URL mentioned in the file Jenkinsfile.

Stage2: Build the application using maven command "*mvn clean package*".

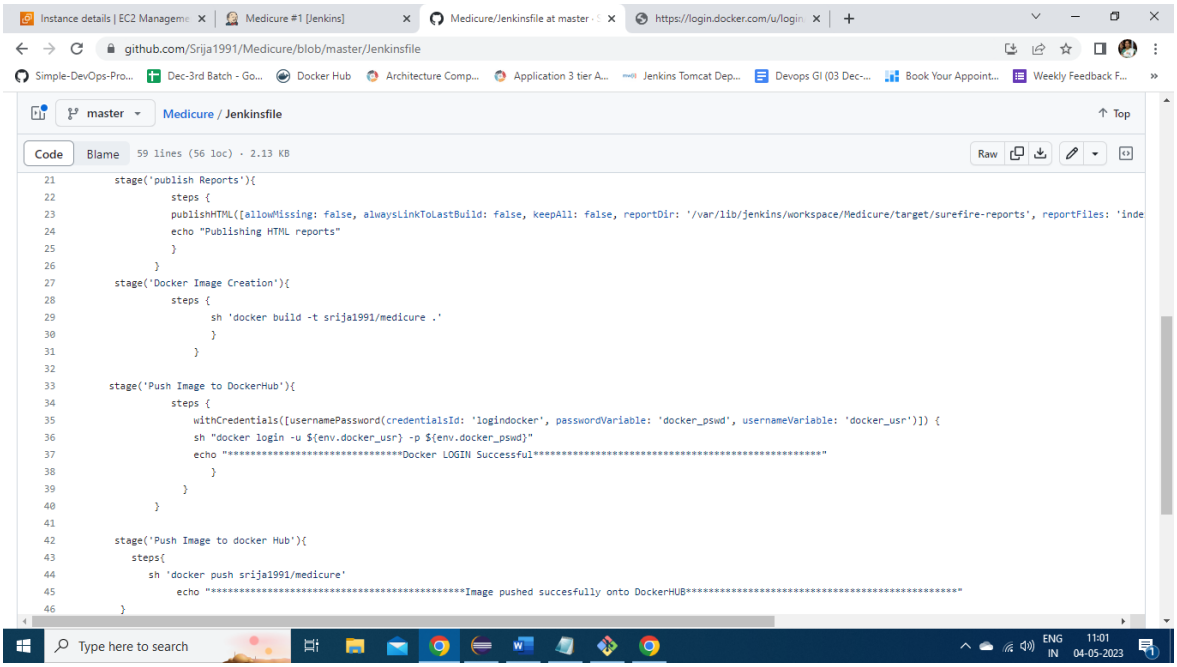
Stage 3: Publish the test reports using HTML publisher.



Stage 4: Create docker image using the Dockerfile and jar file created from build success step.

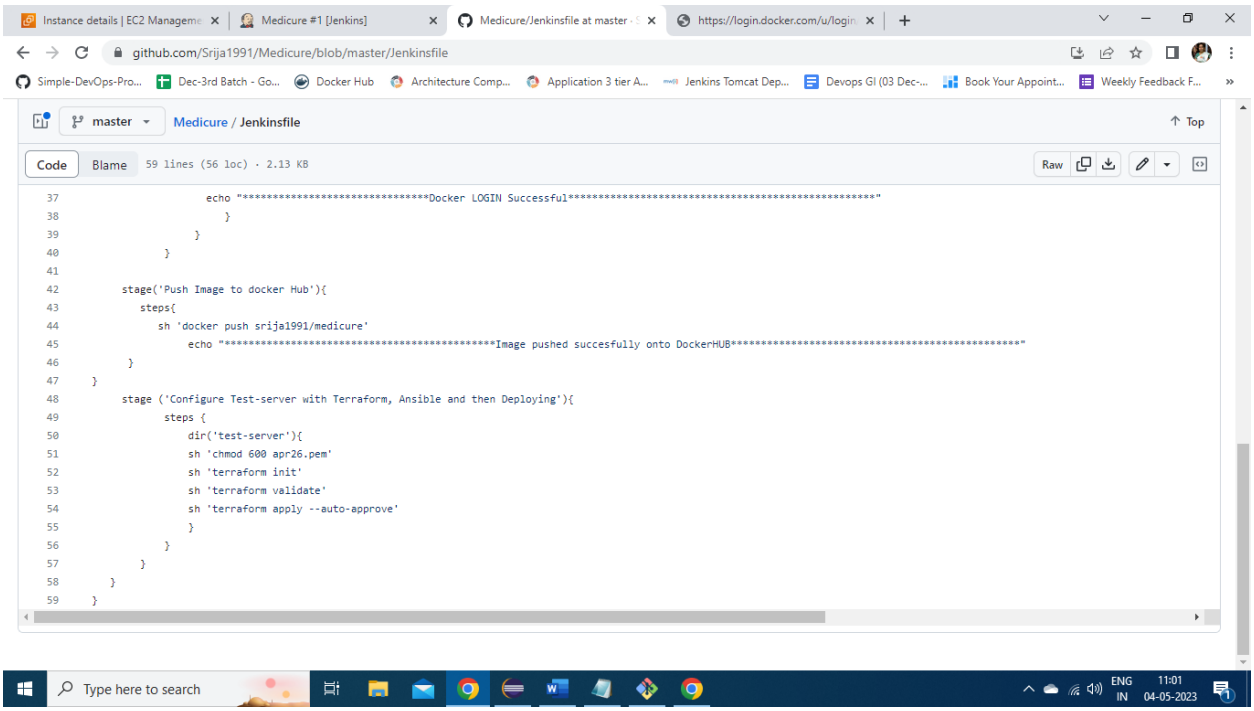


Stage 5: Push the docker mage onto docker hub using appropriate credentials with the name *srija1991/medicure*.

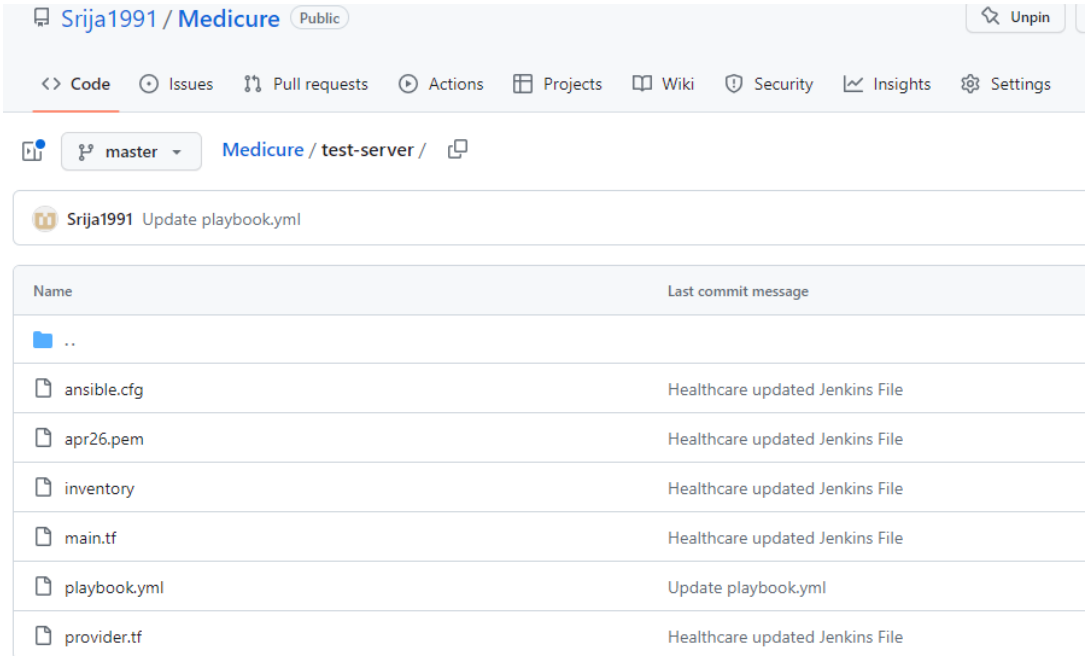


```
stage('publish Reports'){
        steps {
        publishHTML([allowMissing: false, alwaysLinkToLastBuild: false, keepAll: false, reportDir: '/var/lib/jenkins/workspace/Medicure/target/surefire-reports', reportFiles: 'inde
        echo "Publishing HTML reports"
        }
      }
stage('Docker Image Creation'){
        steps {
              sh 'docker build -t srija1991/medicure .'
              }
           }

stage('Push Image to DockerHub'){
        steps {
            withCredentials([usernamePassword(credentialsId: 'logindocker', passwordVariable: 'docker_pswd', usernameVariable: 'docker_usr')]) {
            sh "docker login -u ${env.docker_usr} -p ${env.docker_pswd}"
            echo "******************************Docker LOGIN Successful**********************************************************"
               }
            }
         }

 stage('Push Image to docker Hub'){
    steps{
       sh 'docker push srija1991/medicure'
            echo "*********************************************Image pushed succesfully onto DockerHUB**********************************************"
  }
```

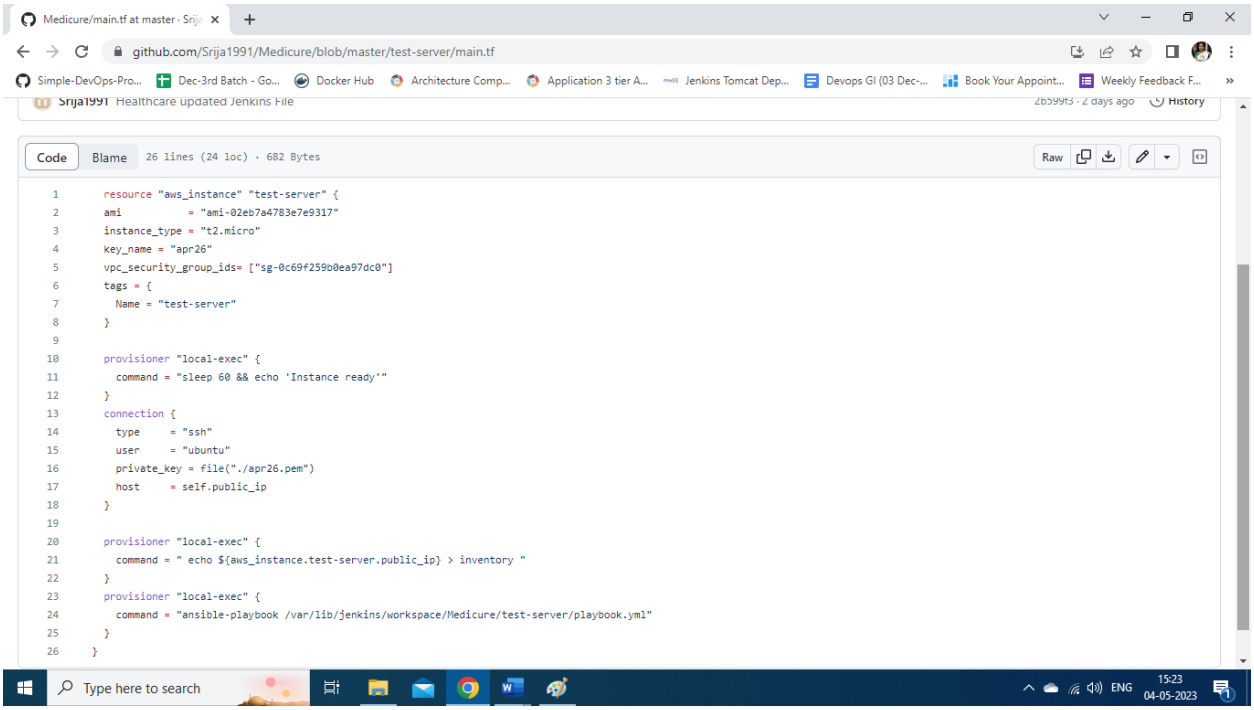
Now stage 6 is used to create an EC2 instance which acts as a test -server. Here is test-server folder which has *main.tf* file to create an EC2 instance in the region mentioned and attach the keypair *"apr26"*.



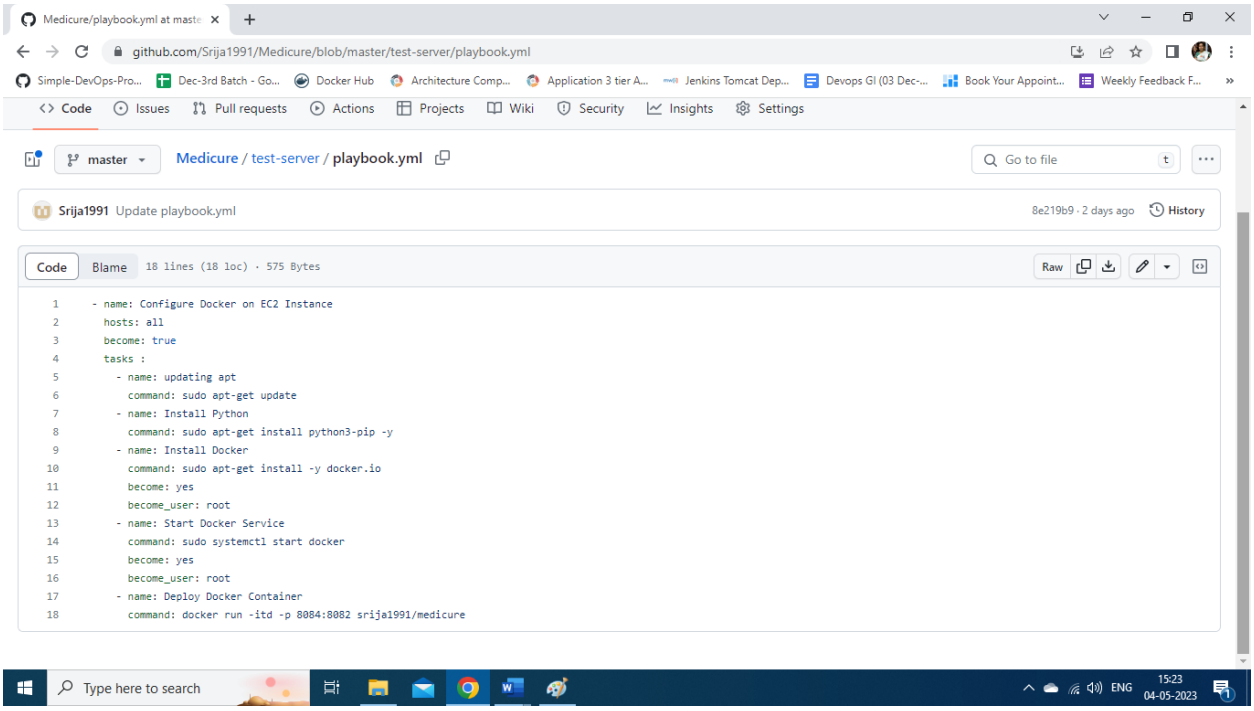
```
                echo "******************************Docker LOGIN Successful**********************************************************"
                   }
                }
             }

      stage('Push Image to docker Hub'){
         steps{
            sh 'docker push srija1991/medicure'
                 echo "*********************************************Image pushed succesfully onto DockerHUB**********************************************"
       }
  }
      stage ('Configure Test-server with Terraform, Ansible and then Deploying'){
            steps {
                dir('test-server'){
                sh 'chmod 600 apr26.pem'
                sh 'terraform init'
                sh 'terraform validate'
                sh 'terraform apply --auto-approve'
                }
             }
          }
      }
  }
```



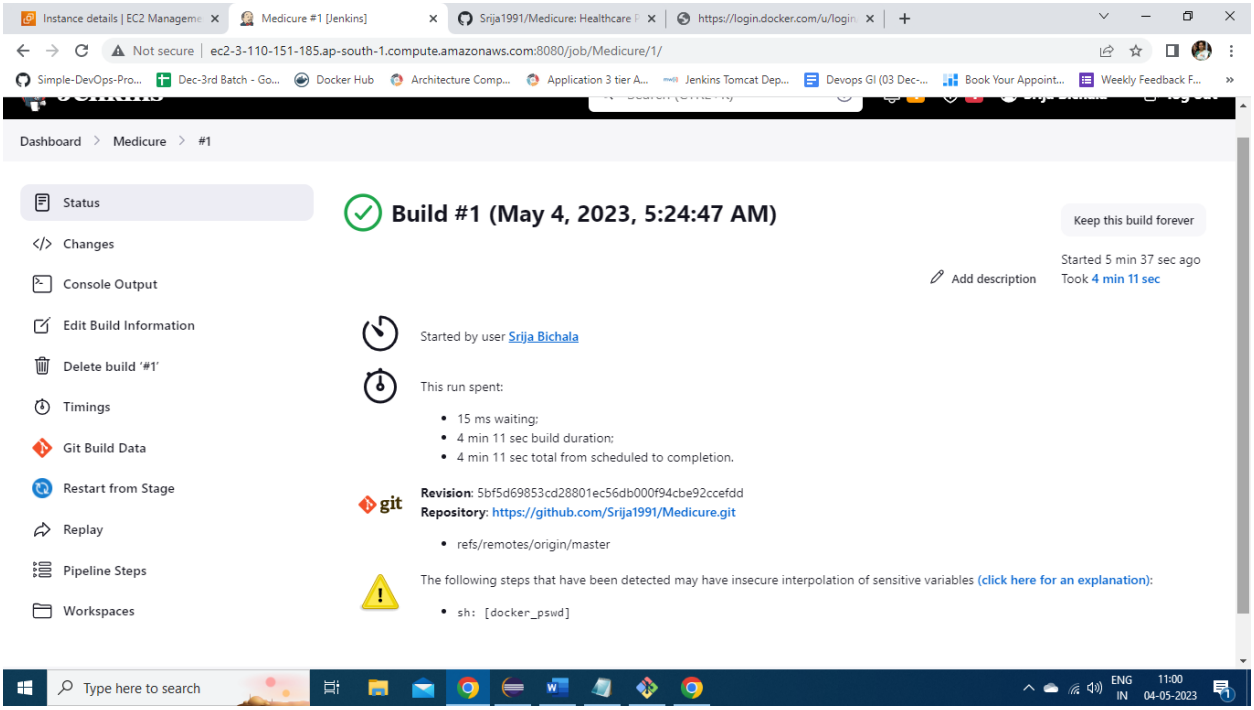
Main.tf file with the terraform script for creating test-server and run ansible playbook to install necessary packages to deploy application onto it.



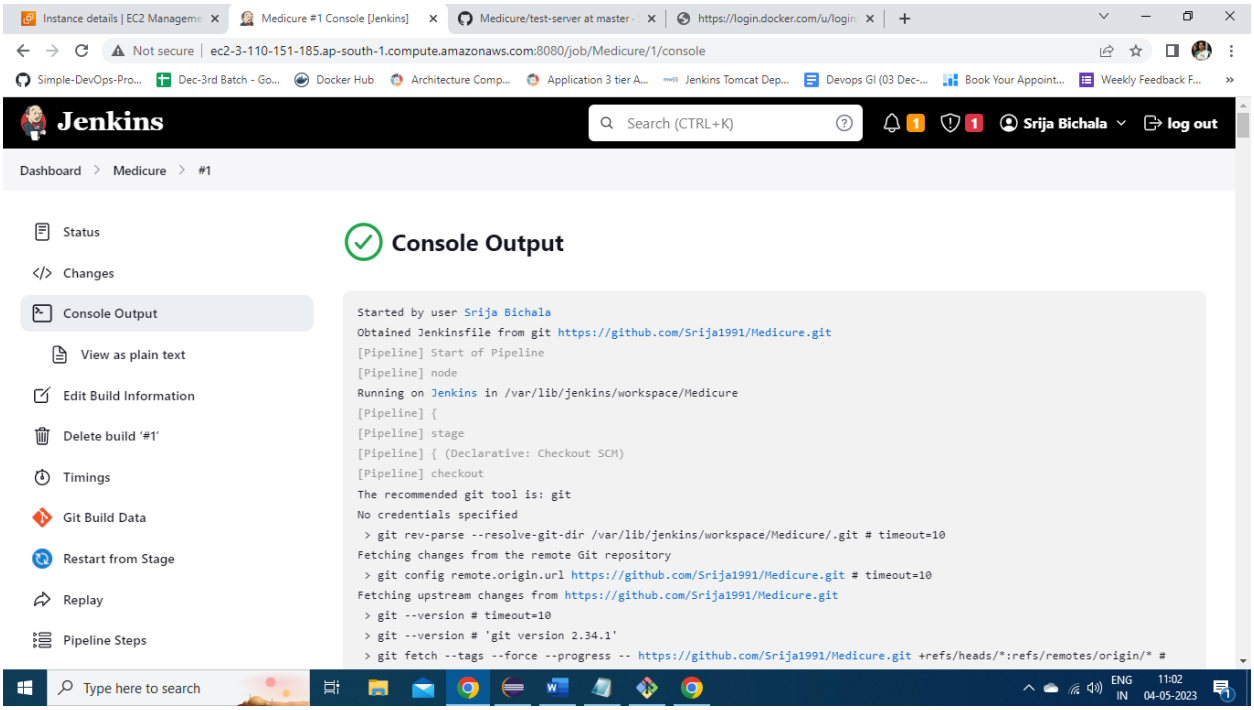
Here is *playbook.yml* to configure the test-server created and deploy the application on port number 8084.



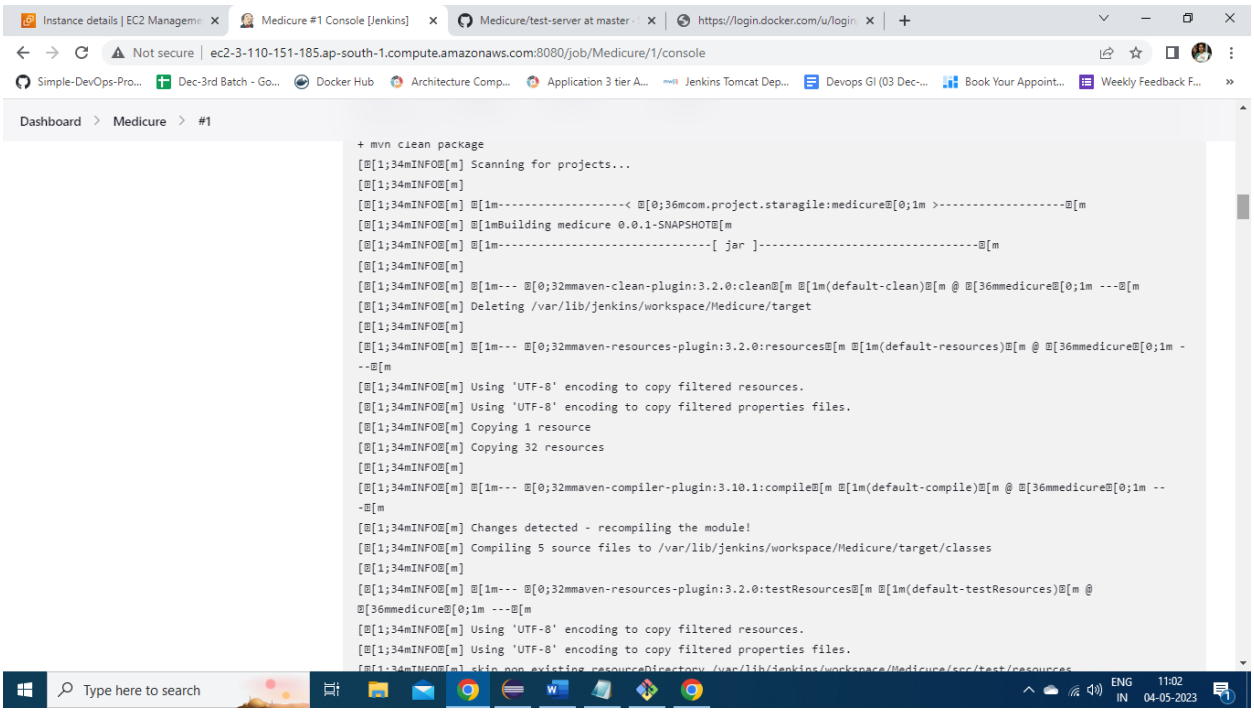
Now the job is build sucessfully and here is console output .

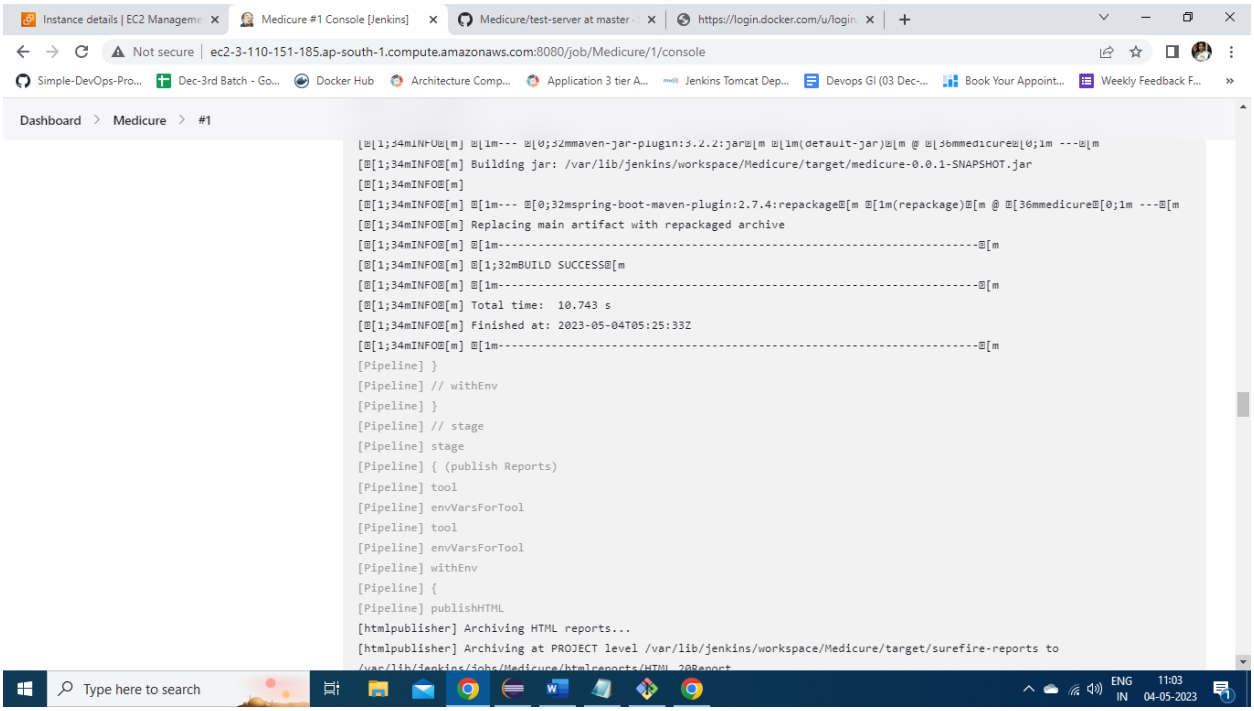


The git checkout is sucessful.



Stage 2 Build the package is successful.





Stage 3: Published the reports to specified folder.

Stage 4: Docker image is created.

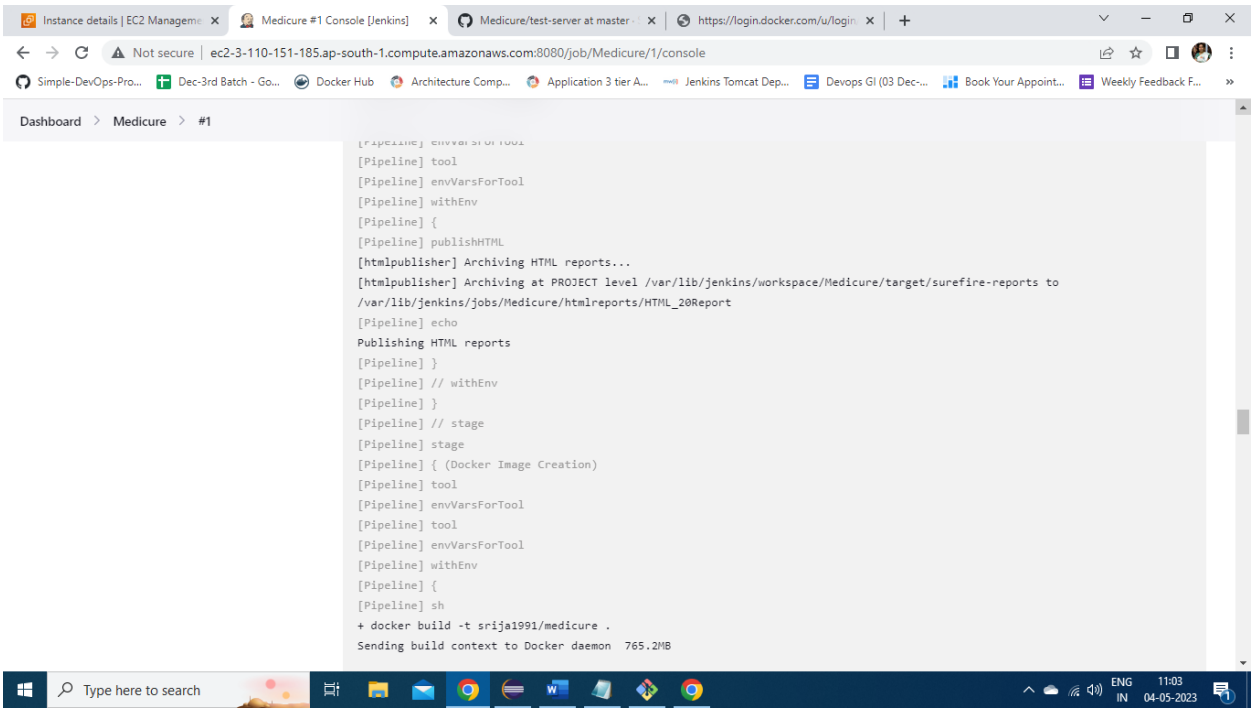
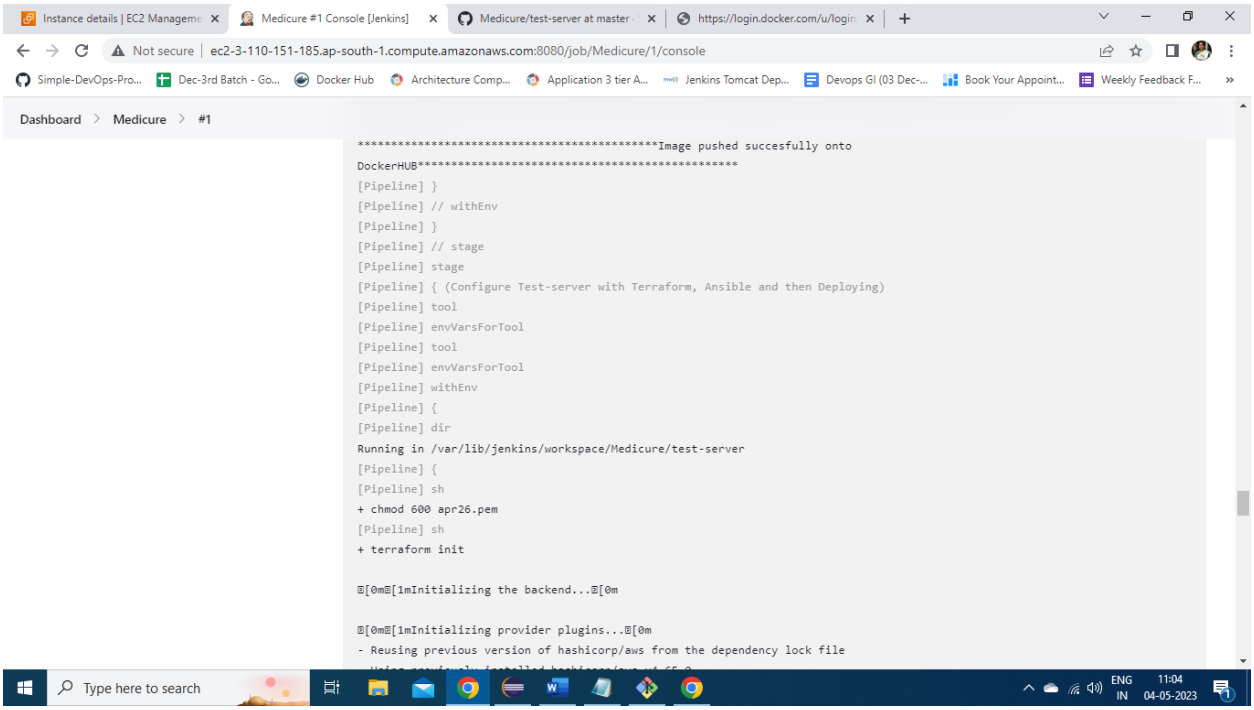
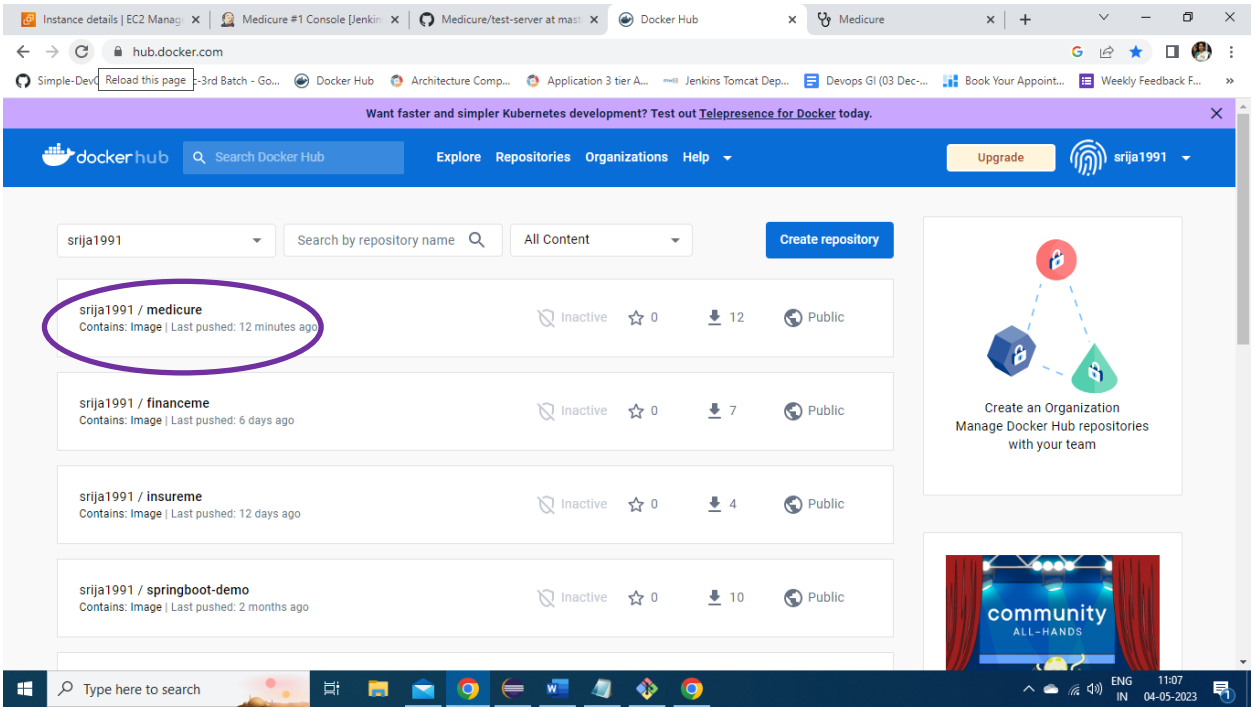
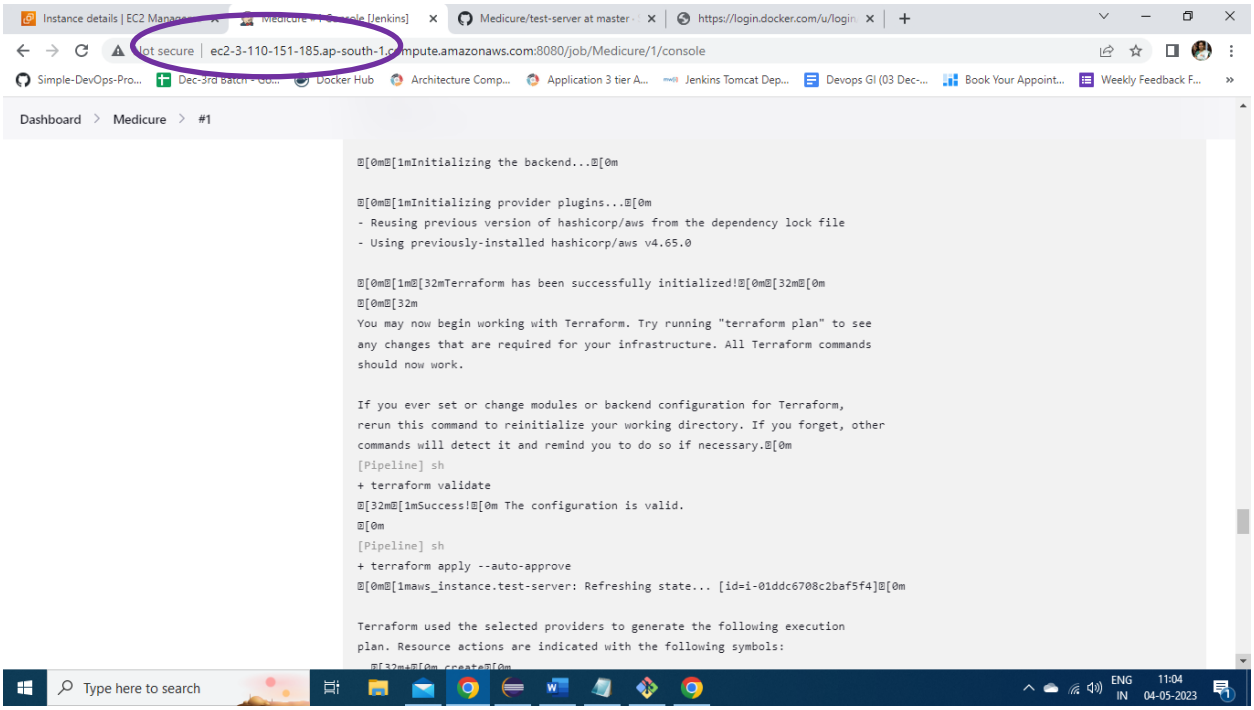


Image is pushed successfully onto docker hub.

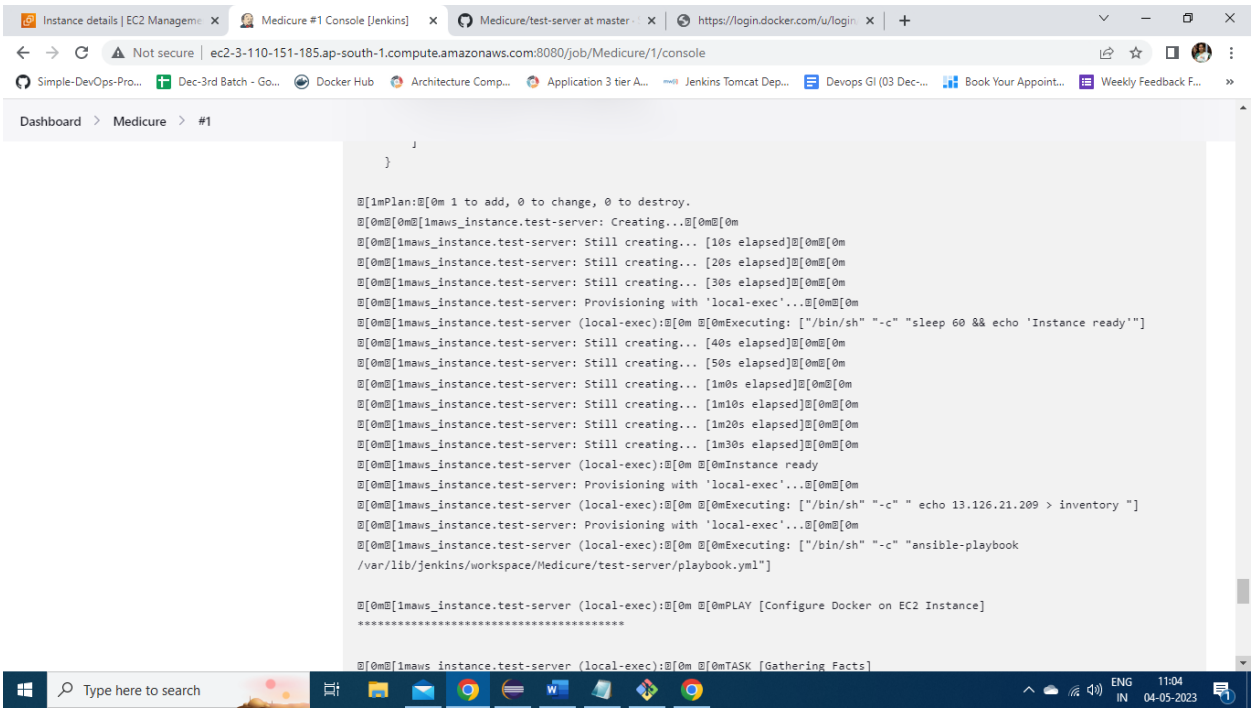


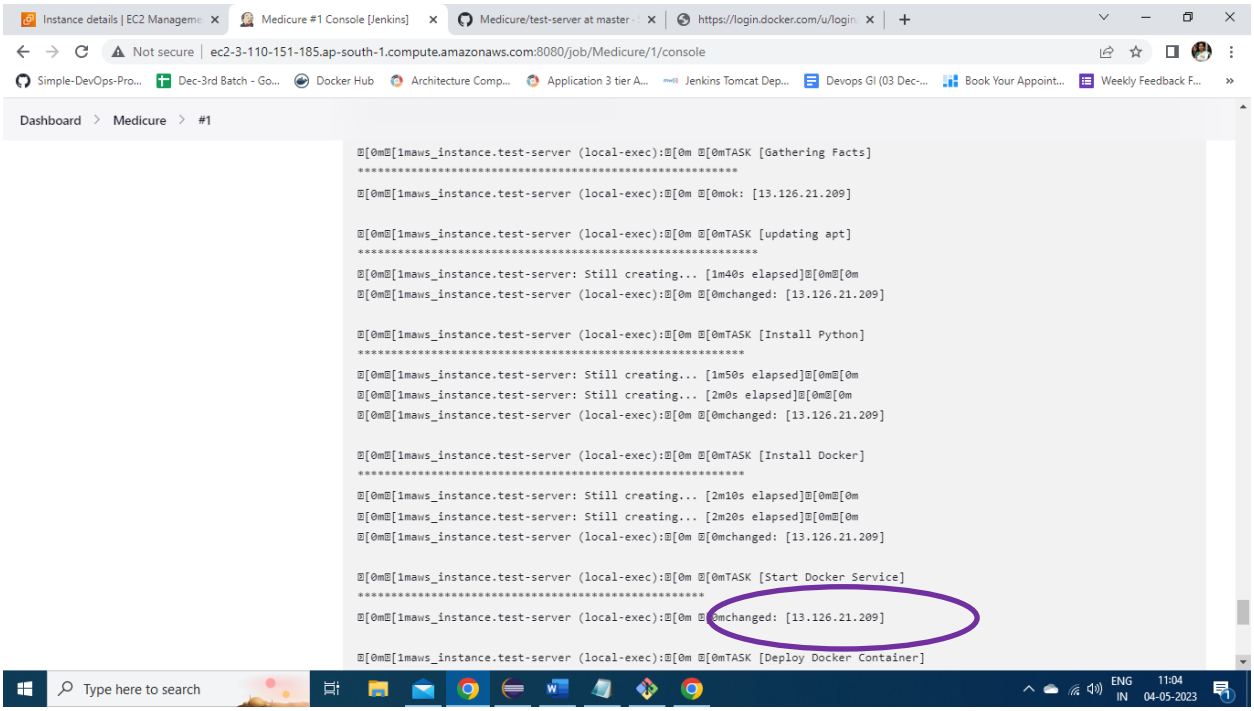


Terraform is initiated and test -server is created with Public IP: 13.126.21.109

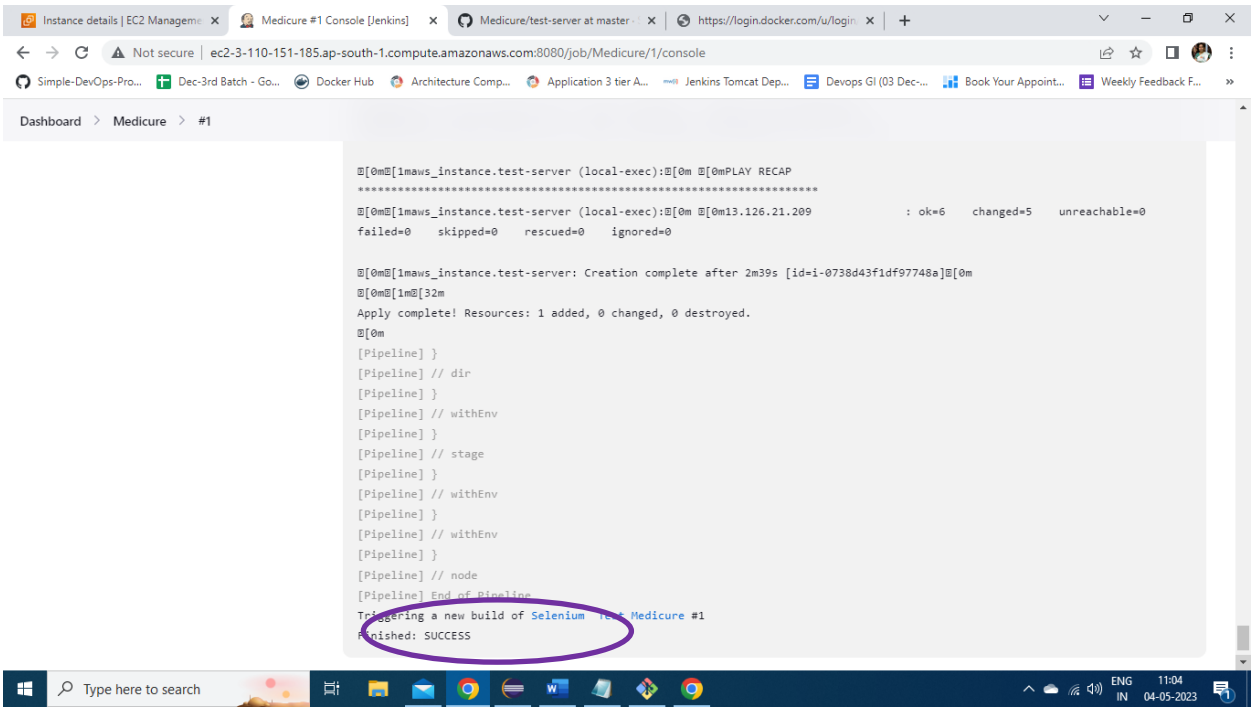


The playbook to configure test-server is started and completed successfully.

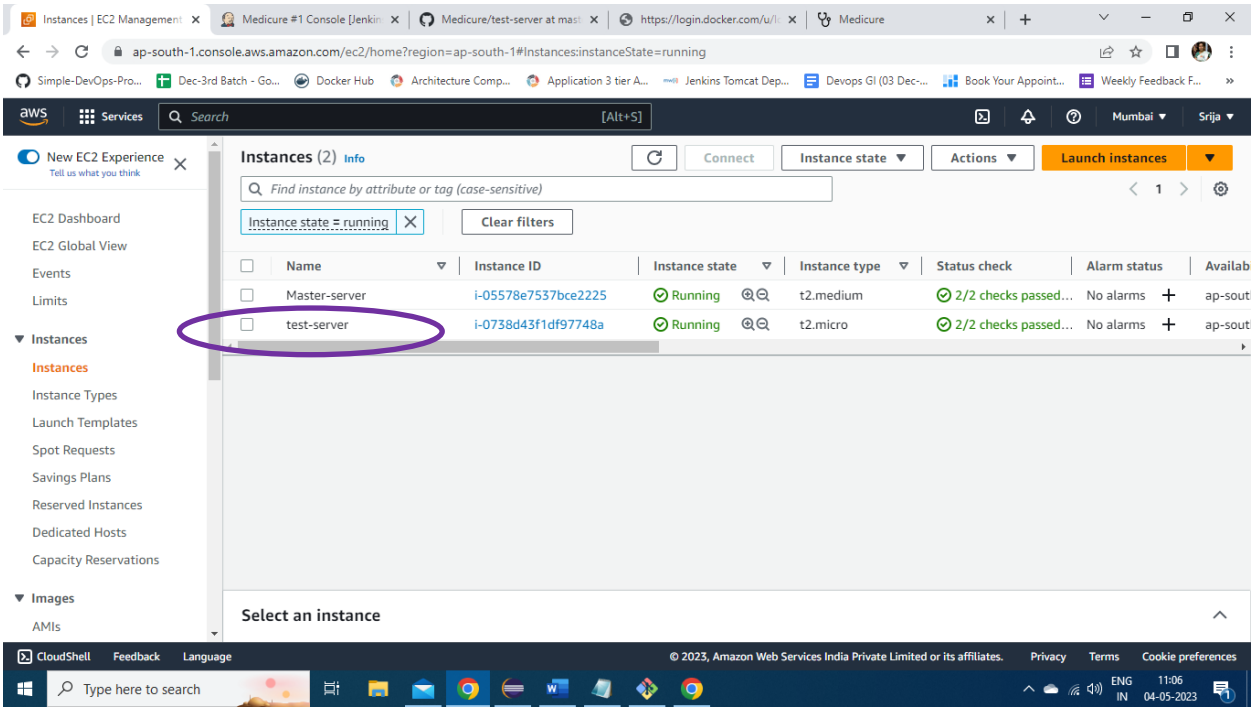


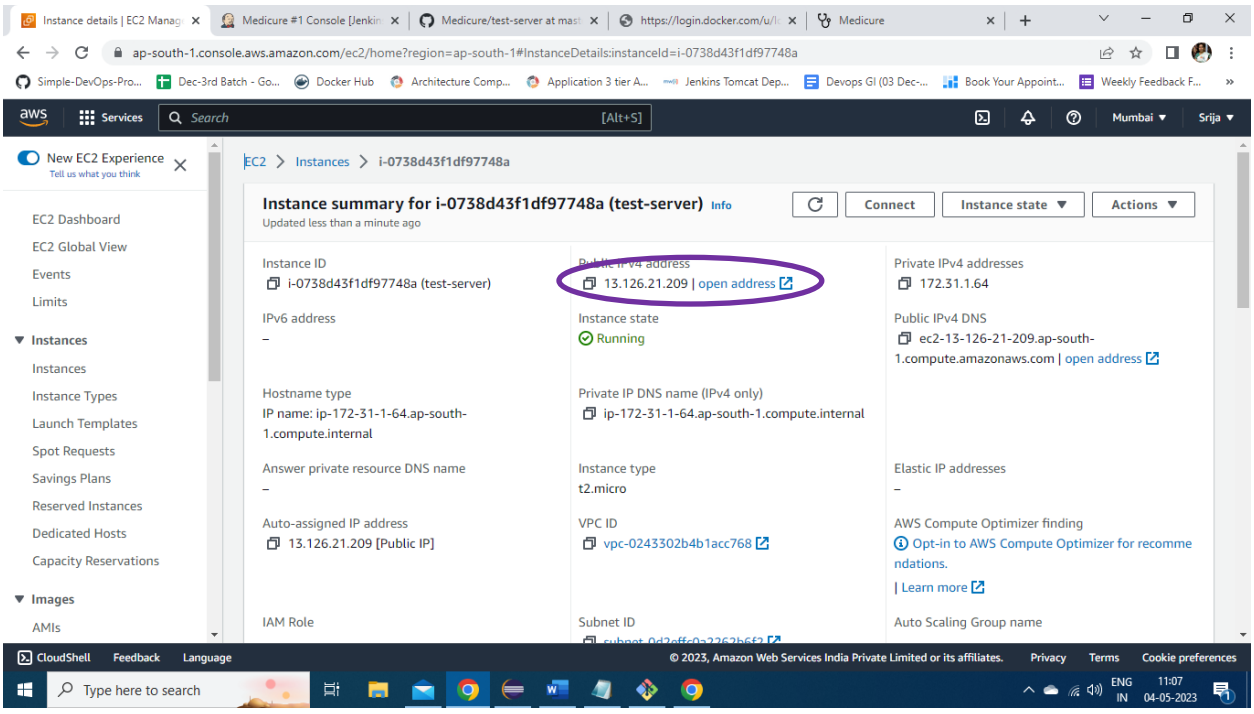


The application is successfully deployed on to the port 8084 and it triggers the selenium test for the server.

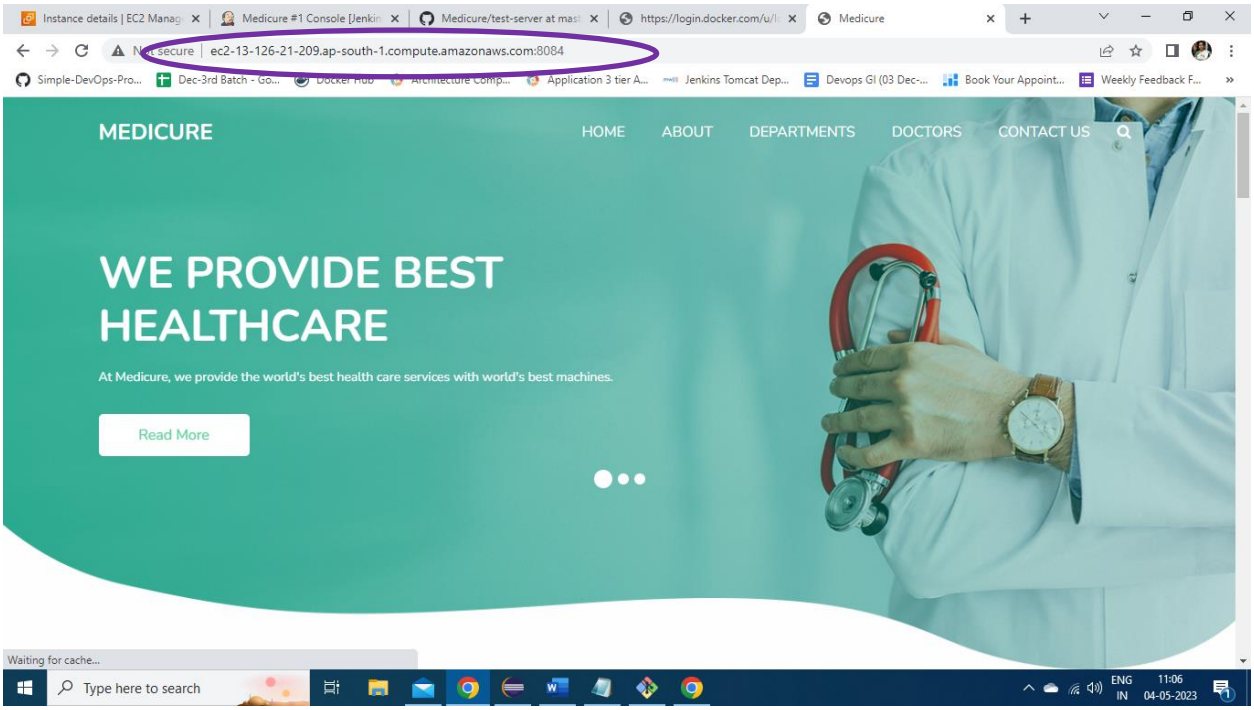


Here is the created test-server using terraform and the IP details of test-server are shown here.

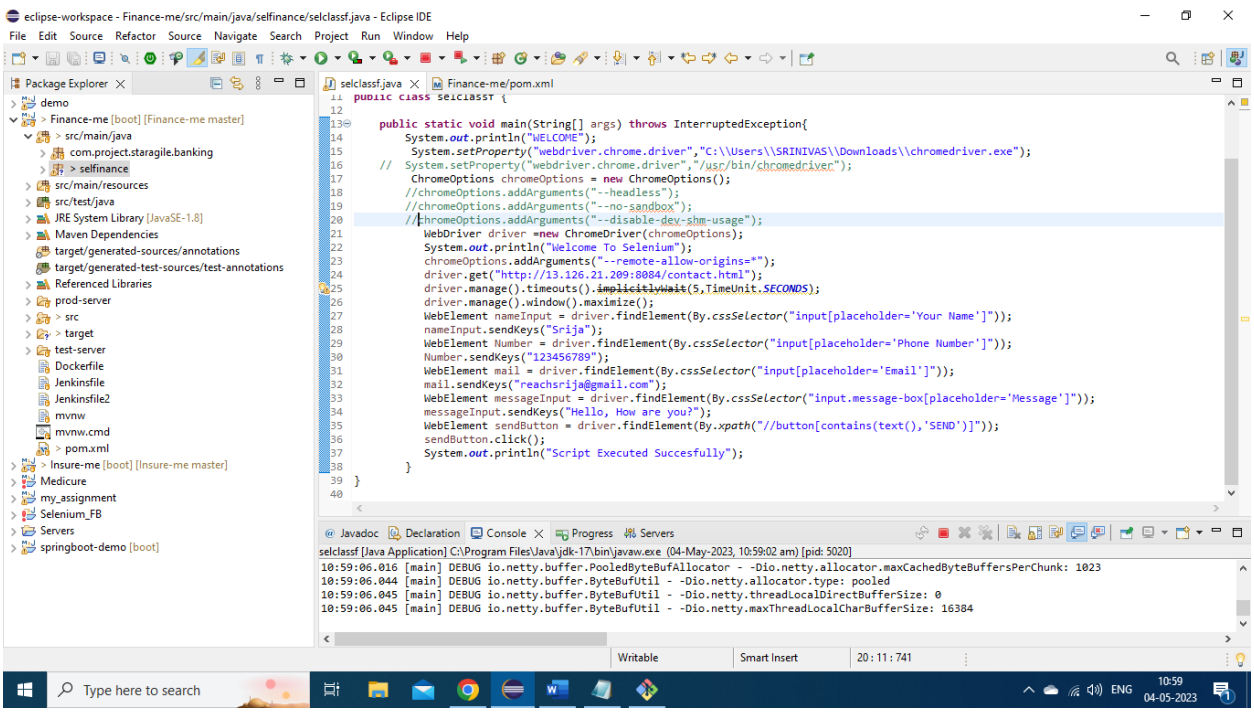




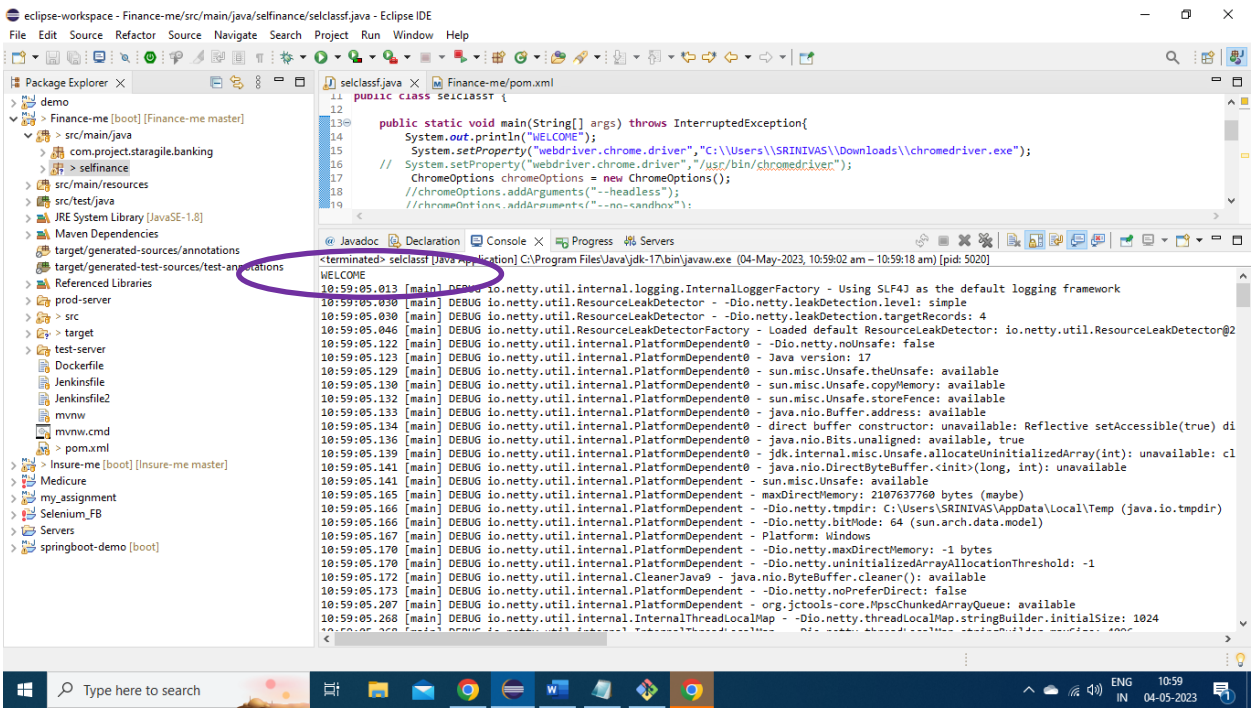
The application on test-server: 13.126.21.109:8084



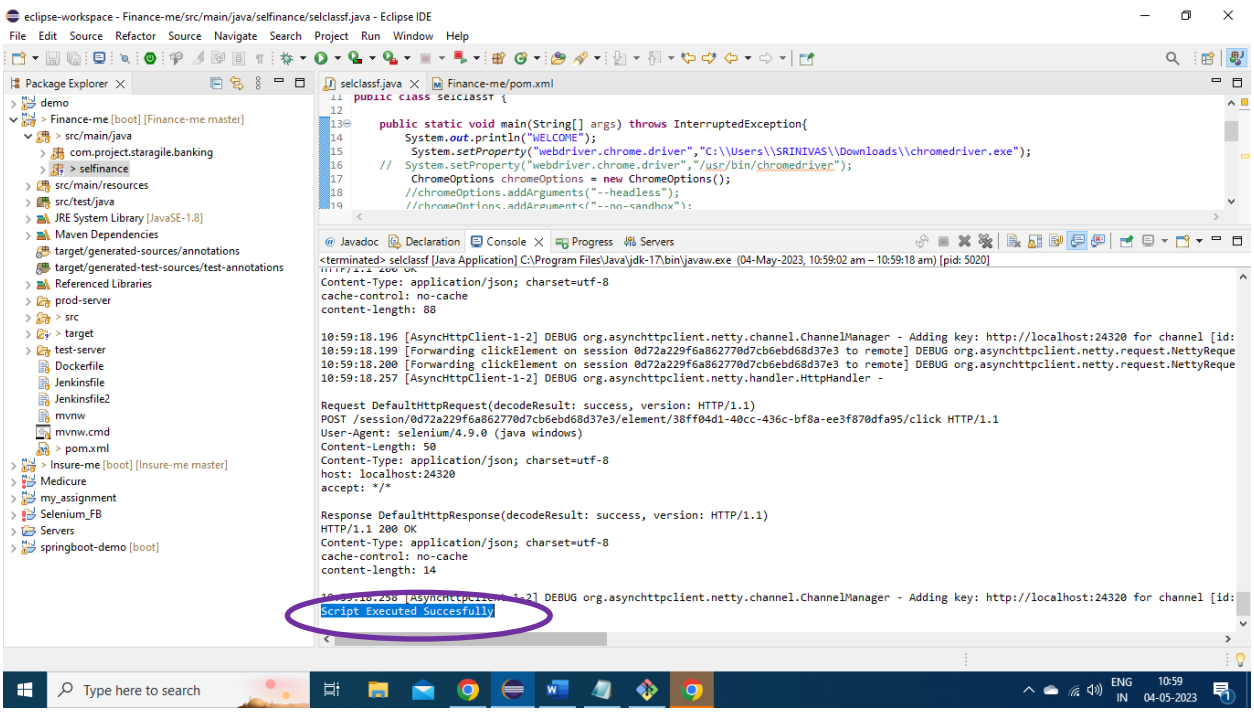
Here is the selenium script written in eclipse to test the deployed application on test-server contacts page.



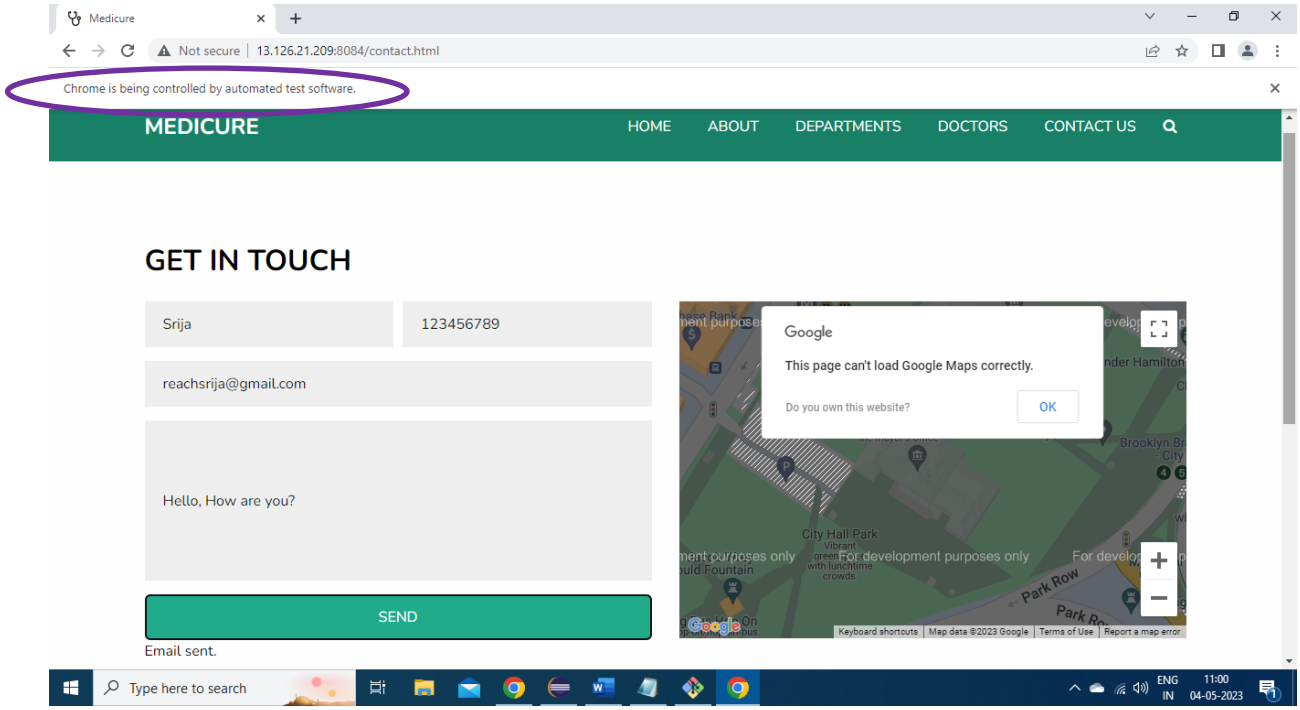
This selenium script is run as java application to test the working of it on windows platform.



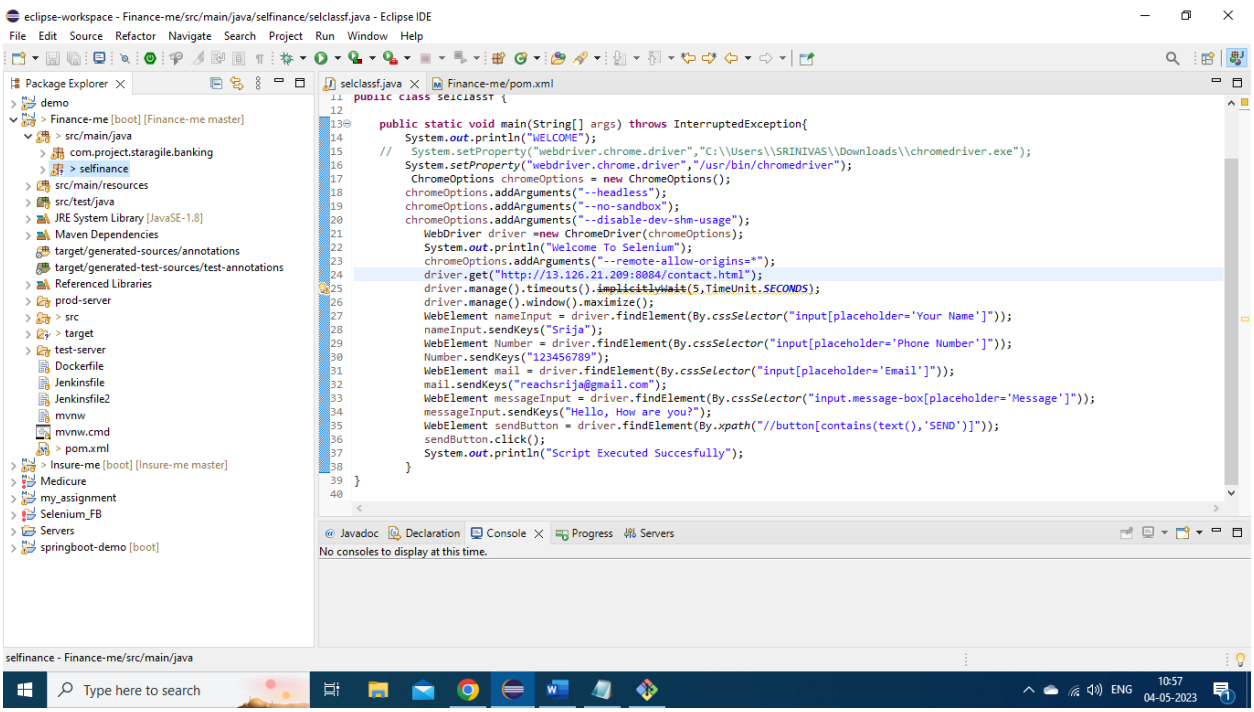
Here is it is successfully started and it automates the application and enters the basic details of a customer willing to contact the Medicure.

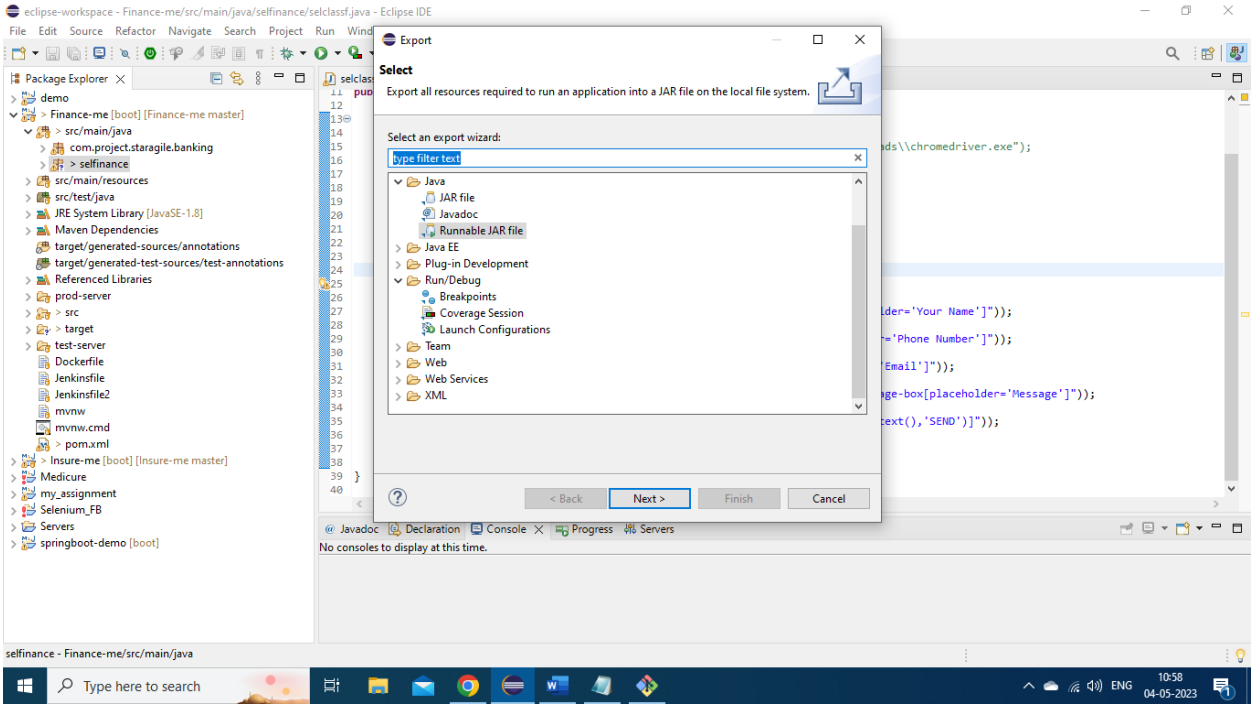


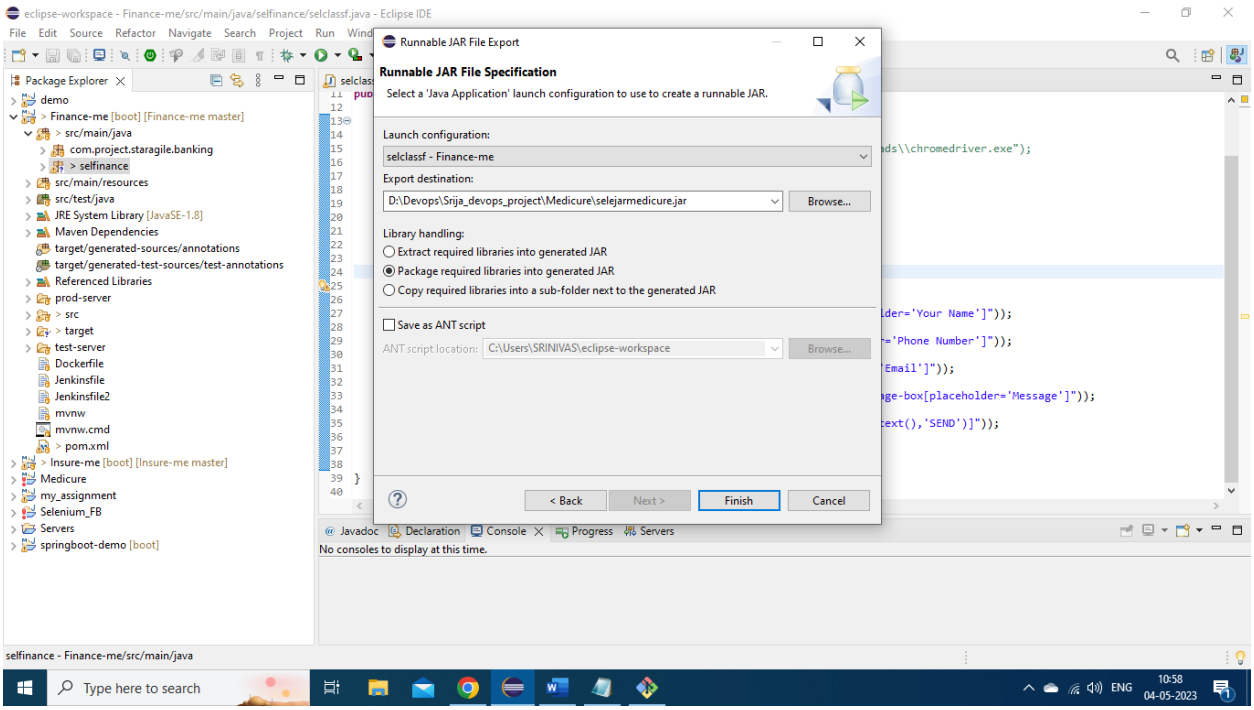
Here is the automated application run though selenium.



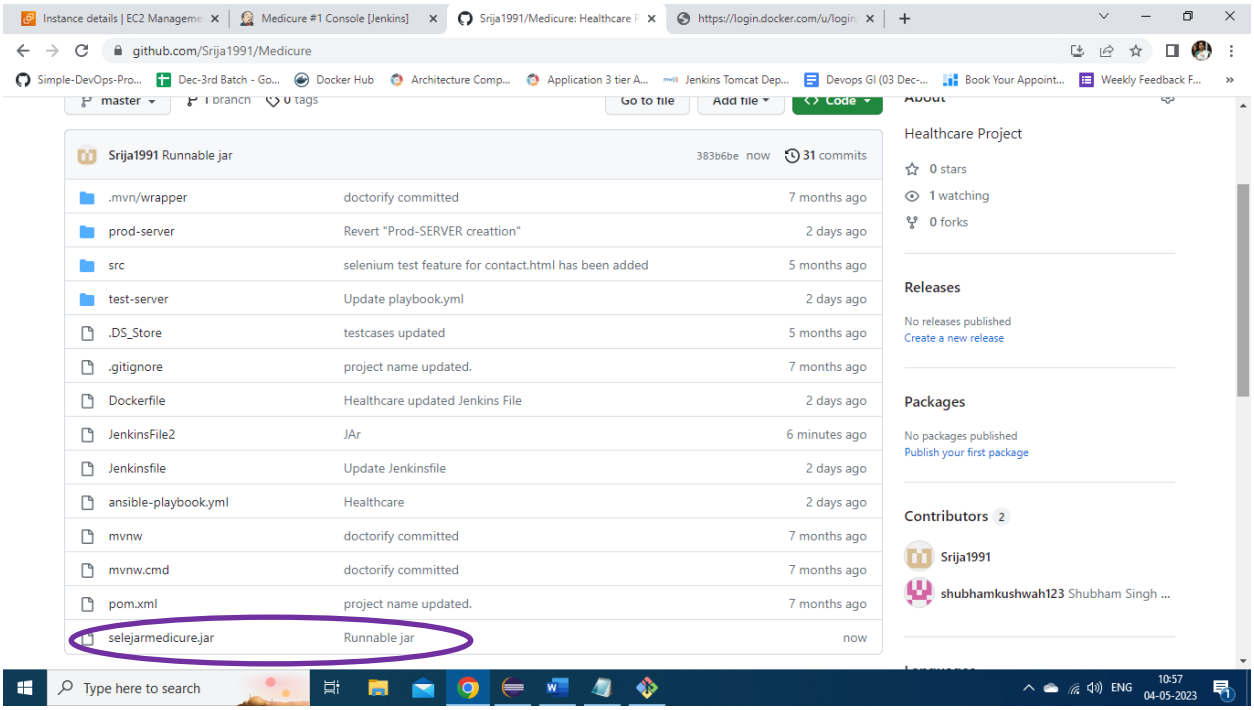
Now the selenium script is modified to run on Linux platform and exported as runnable jar to git hub .



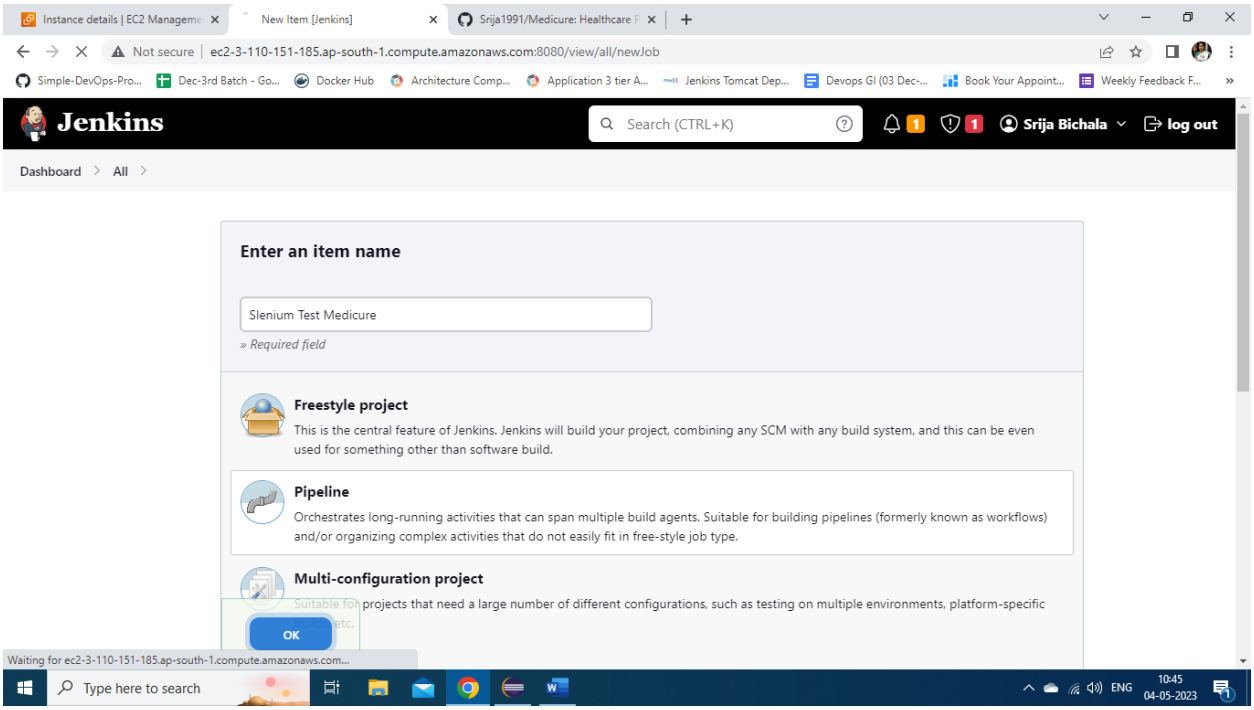

eclipse-workspace - Finance-me/src/main/java/selfinance/selclassf.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run
Package Explorer
demo
Finance-me [boot] [Finance-me master]
src/main/java
com.project.staragile.banking
selfinance
src/main/resources
src/test/java
JRE System Library [JavaSE-1.8]
Maven Dependencies
target/generated-sources/annotations
target/generated-test-sources/test-annotations
Referenced Libraries
prod-server
src
target
test-server
Dockerfile
Jenkinsfile
Jenkinsfile2
mvnw
mvnw.cmd
pom.xml
Insure-me [boot] [Insure-me master]
Medicure
my_assignment
Selenium_FB
Servers
springboot-demo [boot]
Export
Select
Export all resources required to run an application into a JAR file on the local file system.
Select an export wizard:
type filter text
Java
JAR file
Javadoc
Runnable JAR file
Java EE
Plug-in Development
Run/Debug
Breakpoints
Coverage Session
Launch Configurations
Team
Web
Web Services
XML
< Back
Next >
Finish
Cancel
Javadoc Declaration Console Progress Servers
No consoles to display at this time.
selfinance - Finance-me/src/main/java
Type here to search
ENG
10:58
04-05-2023

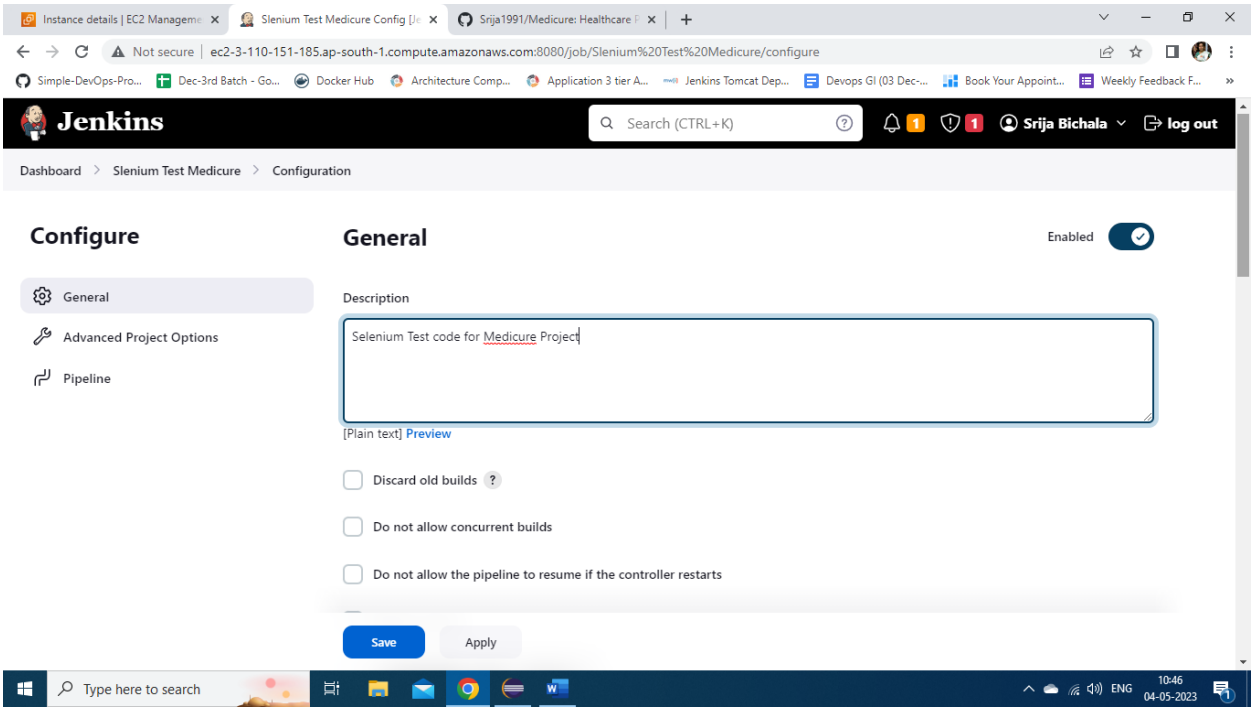

eclipse-workspace - Finance-me/src/main/java/selfinance/selclassf.java - Eclipse IDE
Runnable JAR File Export
Runnable JAR File Specification
Select a 'Java Application' launch configuration to use to create a runnable JAR.
Launch configuration:
selclassf - Finance-me
Export destination:
D:\Devops\Srija_devops_project\Medicure\selejarmedicure.jar
Browse...
Library handling:
Extract required libraries into generated JAR
Package required libraries into generated JAR
Copy required libraries into a sub-folder next to the generated JAR
Save as ANT script
ANT script location:
C:\Users\SRINIVAS\eclipse-workspace
Browse...
< Back
Next >
Finish
Cancel
Javadoc Declaration Console Progress Servers
No consoles to display at this time.
selfinance - Finance-me/src/main/java
Type here to search
ENG
10:58
04-05-2023

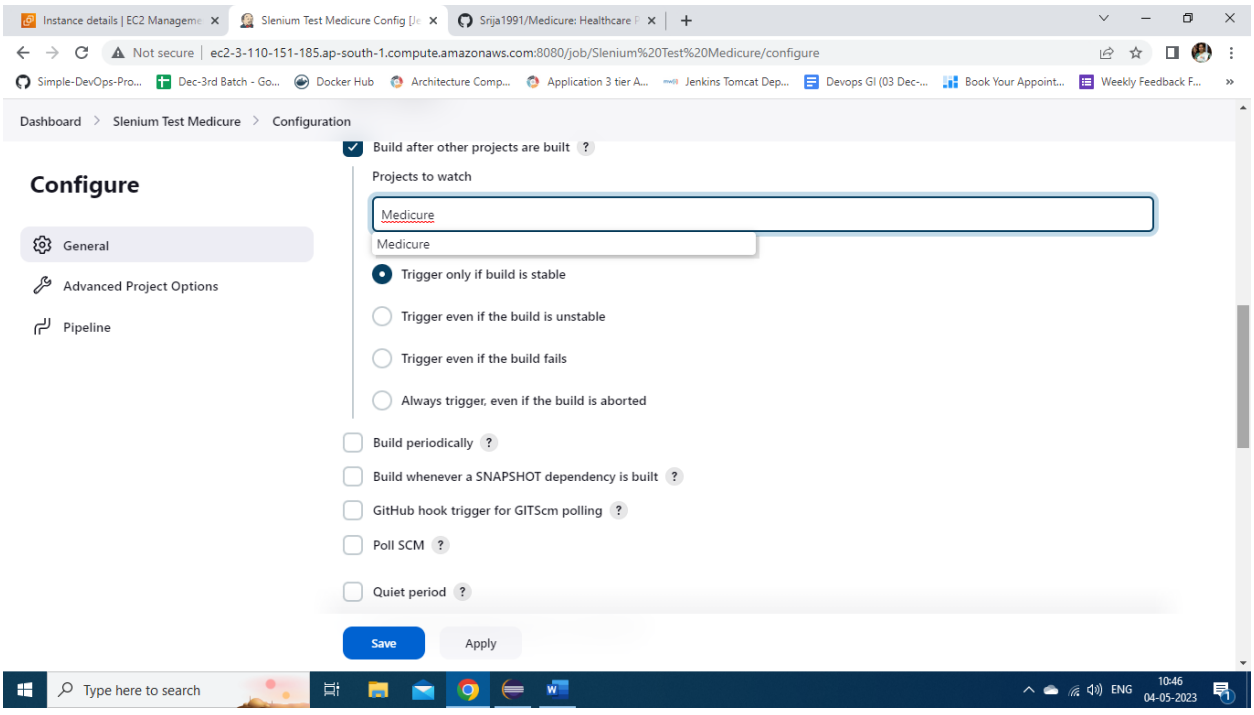
The exported jar is pushed onto GitHub to execute the selenium script on EC2 instance.



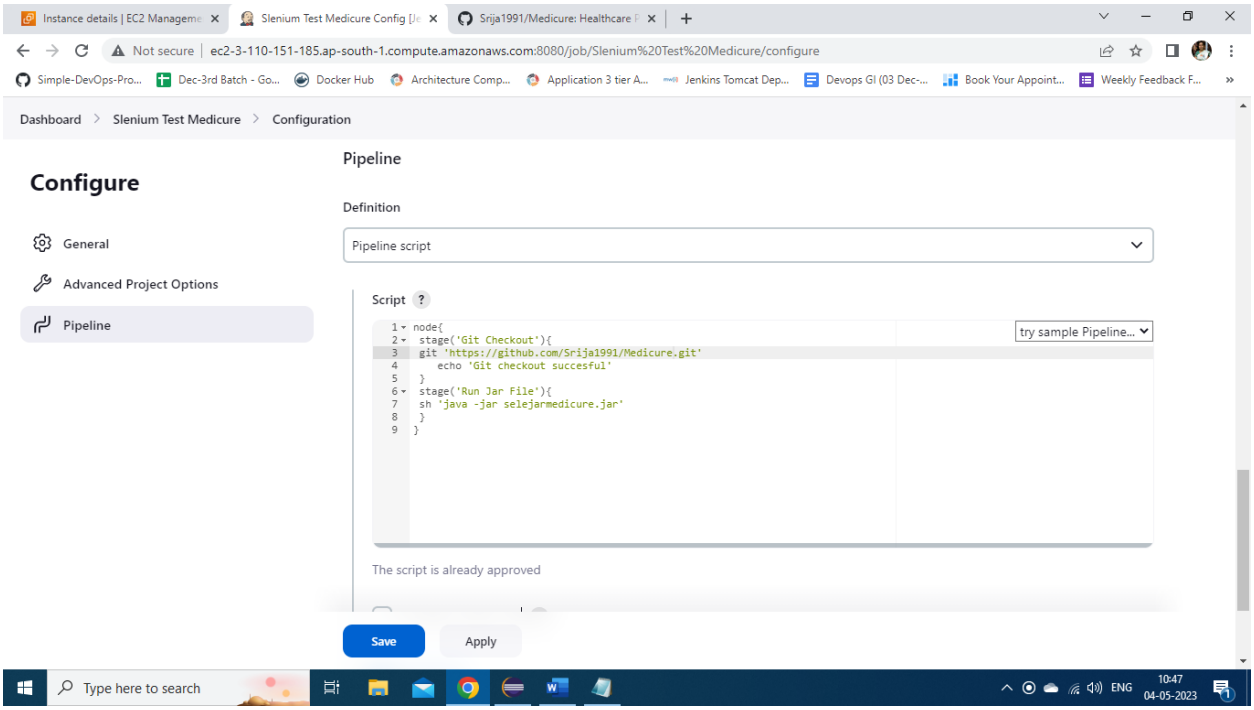
A new job named Selenium Test Medicure is created to run the selenium script after the successful execution of Medicure job.



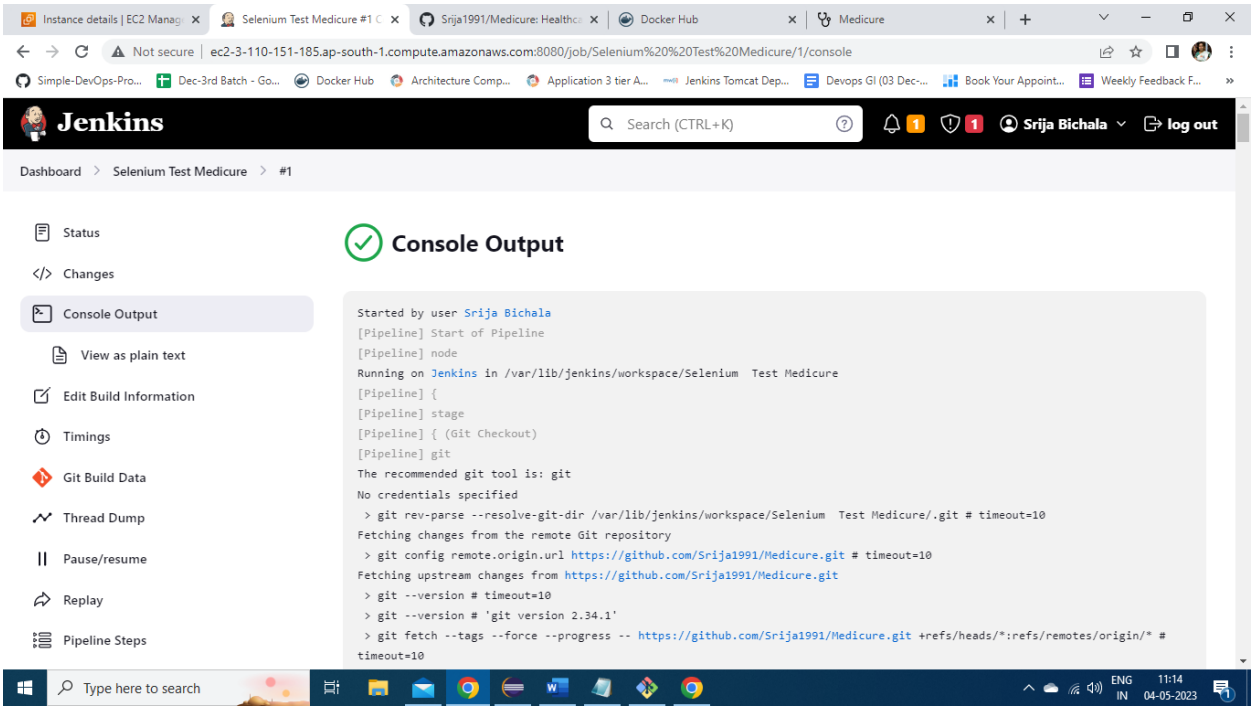

Instance details | EC2 Manageme
Slenium Test Medicure Config [Je
Srija1991/Medicure: Healthcare F
Not secure | ec2-3-110-151-185.ap-south-1.compute.amazonaws.com:8080/job/Slenium%20Test%20Medicure/configure
Simple-DevOps-Pro...
Dec-3rd Batch - Go...
Docker Hub
Architecture Comp...
Application 3 tier A...
Jenkins Tomcat Dep...
Devops GI (03 Dec-...
Book Your Appoint...
Weekly Feedback F...
Jenkins
Search (CTRL+K)
Srija Bichala
log out
Dashboard
Slenium Test Medicure
Configuration
Configure
General
Advanced Project Options
Pipeline
General
Enabled
Description
Selenium Test code for Medicure Project
[Plain text] Preview
Discard old builds
Do not allow concurrent builds
Do not allow the pipeline to resume if the controller restarts
Save
Apply
Type here to search
ENG
10:46
04-05-2023

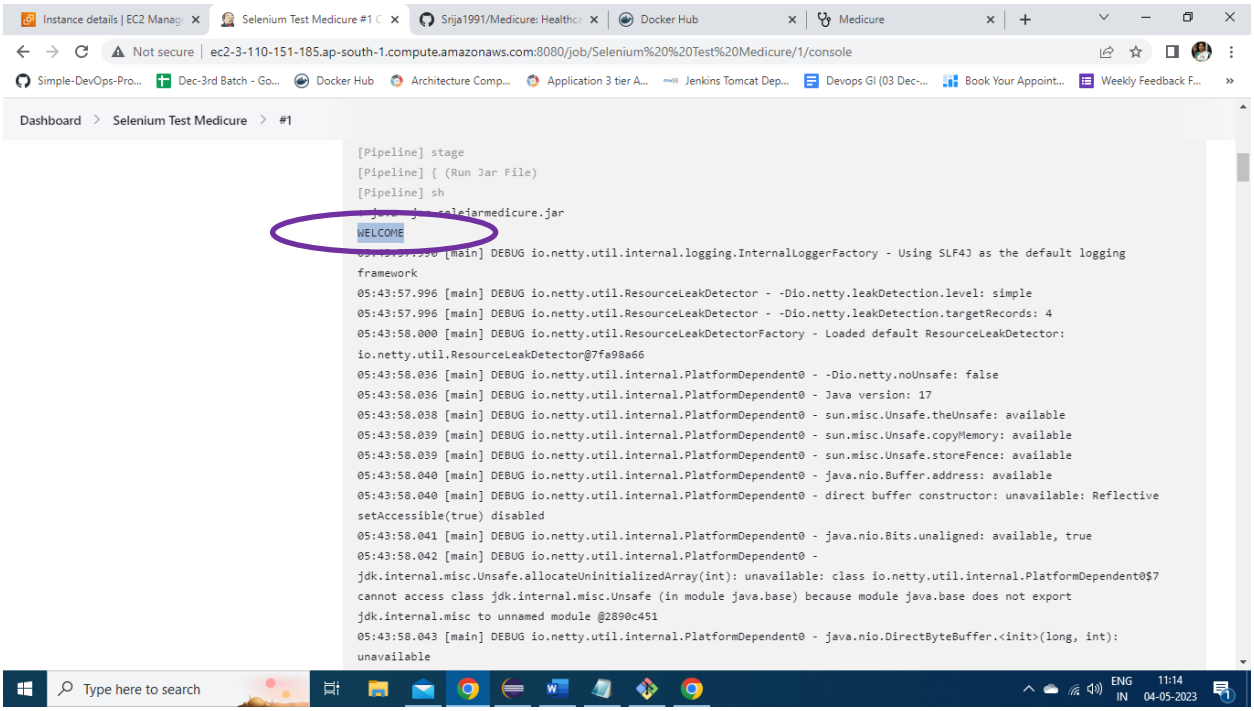

Instance details | EC2 Manageme
Slenium Test Medicure Config [Je
Srija1991/Medicure: Healthcare F
Not secure | ec2-3-110-151-185.ap-south-1.compute.amazonaws.com:8080/job/Slenium%20Test%20Medicure/configure
Dashboard
Slenium Test Medicure
Configuration
Configure
General
Advanced Project Options
Pipeline
Build after other projects are built
Projects to watch
Medicure
Medicure
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Always trigger, even if the build is aborted
Build periodically
Build whenever a SNAPSHOT dependency is built
GitHub hook trigger for GITScm polling
Poll SCM
Quiet period
Save
Apply
Type here to search
ENG
10:46
04-05-2023

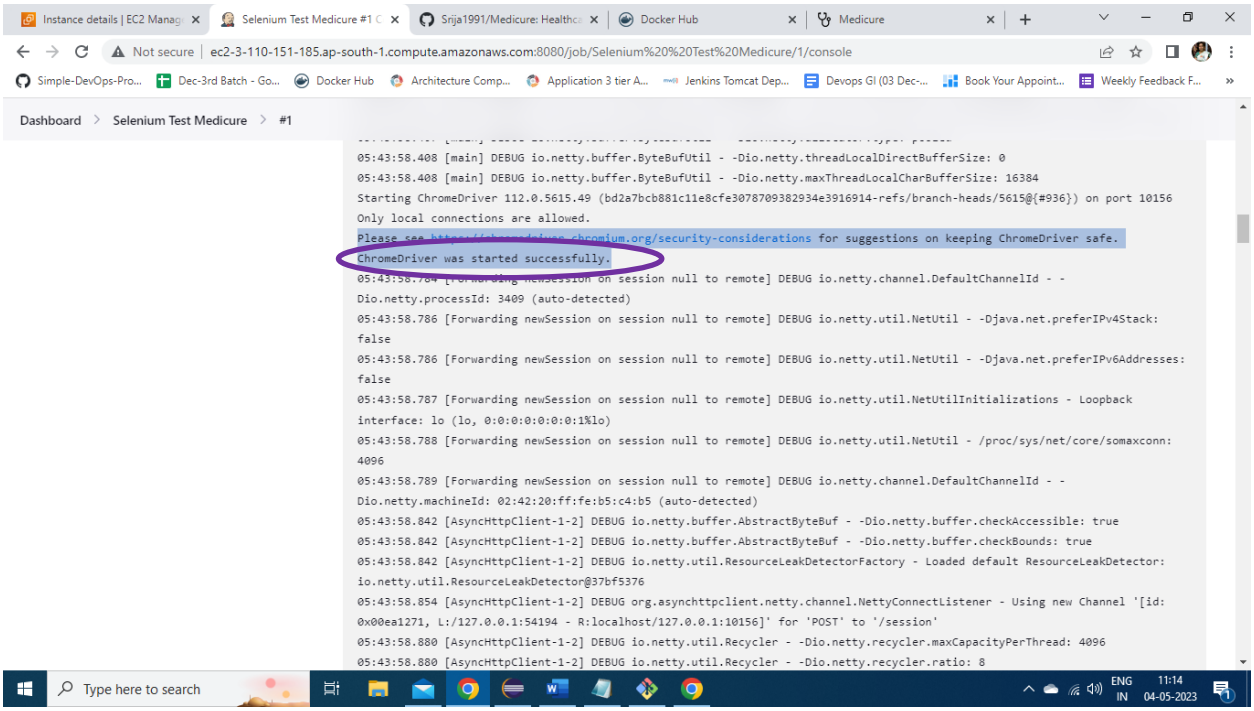
Here is the pipeline script to execute the jar on EC2 instance.

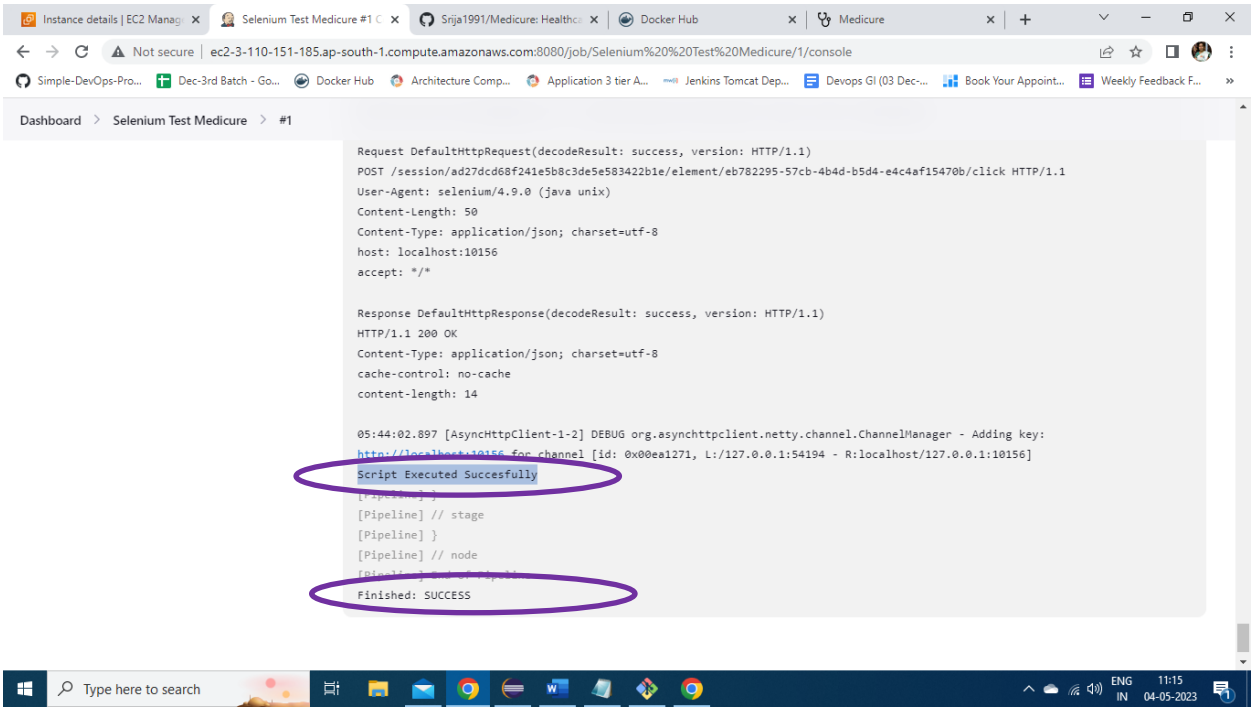


The console output for this job is shown here. It shows the chrome is started successfully and script is executed successfully.

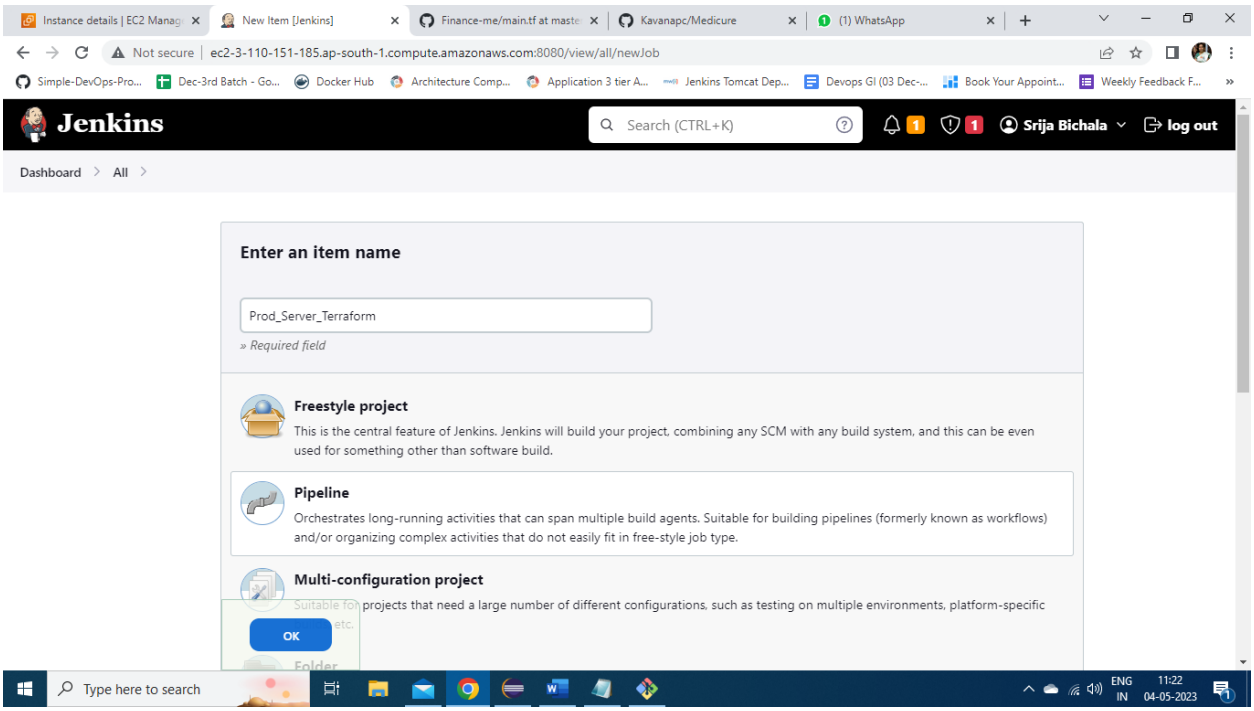


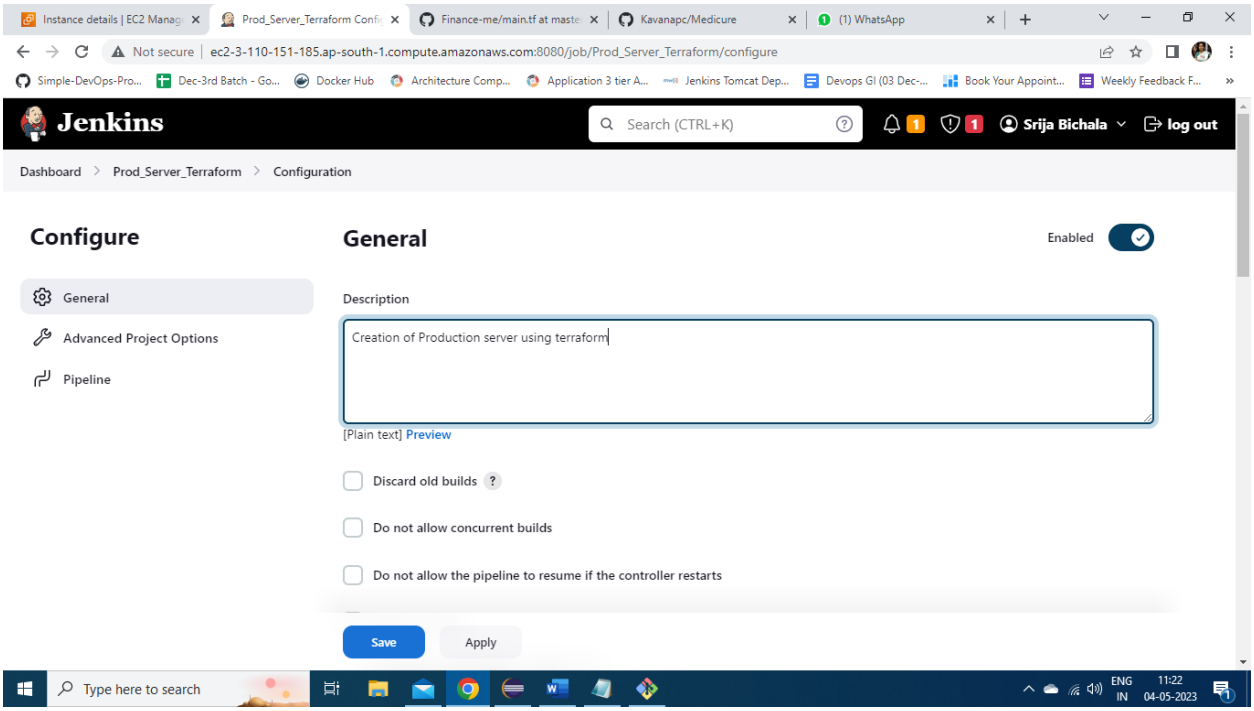

Instance details | EC2 Manag
Selenium Test Medicure #1 C
Srija1991/Medicure: Healthc
Docker Hub
Medicure
Not secure | ec2-3-110-151-185.ap-south-1.compute.amazonaws.com:8080/job/Selenium%20%20Test%20Medicure/1/console
Simple-DevOps-Pro...
Dec-3rd Batch - Go...
Docker Hub
Architecture Comp...
Application 3 tier A...
Jenkins Tomcat Dep...
Devops GI (03 Dec-...
Book Your Appoint...
Weekly Feedback F...
Dashboard > Selenium Test Medicure > #1
[Pipeline] stage
[Pipeline] { (Run Jar File)
[Pipeline] sh
+ java -jar seleiarmedicure.jar
WELCOME
framework
05:43:57.996 [main] DEBUG io.netty.util.ResourceLeakDetector - -Dio.netty.leakDetection.level: simple
05:43:57.996 [main] DEBUG io.netty.util.ResourceLeakDetector - -Dio.netty.leakDetection.targetRecords: 4
05:43:58.000 [main] DEBUG io.netty.util.ResourceLeakDetectorFactory - Loaded default ResourceLeakDetector:
io.netty.util.ResourceLeakDetector@7fa98a66
05:43:58.036 [main] DEBUG io.netty.util.internal.PlatformDependent0 - -Dio.netty.noUnsafe: false
05:43:58.036 [main] DEBUG io.netty.util.internal.PlatformDependent0 - Java version: 17
05:43:58.038 [main] DEBUG io.netty.util.internal.PlatformDependent0 - sun.misc.Unsafe.theUnsafe: available
05:43:58.039 [main] DEBUG io.netty.util.internal.PlatformDependent0 - sun.misc.Unsafe.copyMemory: available
05:43:58.039 [main] DEBUG io.netty.util.internal.PlatformDependent0 - sun.misc.Unsafe.storeFence: available
05:43:58.040 [main] DEBUG io.netty.util.internal.PlatformDependent0 - java.nio.Buffer.address: available
05:43:58.040 [main] DEBUG io.netty.util.internal.PlatformDependent0 - direct buffer constructor: unavailable: Reflective
setAccessible(true) disabled
05:43:58.041 [main] DEBUG io.netty.util.internal.PlatformDependent0 - java.nio.Bits.unaligned: available, true
05:43:58.042 [main] DEBUG io.netty.util.internal.PlatformDependent0 -
jdk.internal.misc.Unsafe.allocateUninitializedArray(int): unavailable: class io.netty.util.internal.PlatformDependent0$7
cannot access class jdk.internal.misc.Unsafe (in module java.base) because module java.base does not export
jdk.internal.misc to unnamed module @2890c451
05:43:58.043 [main] DEBUG io.netty.util.internal.PlatformDependent0 - java.nio.DirectByteBuffer.<init>(long, int):
unavailable
Type here to search
ENG
IN
11:14
04-05-2023


Instance details | EC2 Manag
Selenium Test Medicure #1 C
Srija1991/Medicure: Healthc
Docker Hub
Medicure
Not secure | ec2-3-110-151-185.ap-south-1.compute.amazonaws.com:8080/job/Selenium%20%20Test%20Medicure/1/console
Simple-DevOps-Pro...
Dec-3rd Batch - Go...
Docker Hub
Architecture Comp...
Application 3 tier A...
Jenkins Tomcat Dep...
Devops GI (03 Dec-...
Book Your Appoint...
Weekly Feedback F...
Dashboard > Selenium Test Medicure > #1
05:43:58.408 [main] DEBUG io.netty.buffer.ByteBufUtil - -Dio.netty.threadLocalDirectBufferSize: 0
05:43:58.408 [main] DEBUG io.netty.buffer.ByteBufUtil - -Dio.netty.maxThreadLocalCharBufferSize: 16384
Starting ChromeDriver 112.0.5615.49 (bd2a7bcb881c11e8cfe3078709382934e3916914-refs/branch-heads/5615@{#936}) on port 10156
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Dio.netty.processId: 3409 (auto-detected)
05:43:58.786 [Forwarding newSession on session null to remote] DEBUG io.netty.util.NetUtil - -Djava.net.preferIPv4Stack:
false
05:43:58.786 [Forwarding newSession on session null to remote] DEBUG io.netty.util.NetUtil - -Djava.net.preferIPv6Addresses:
false
05:43:58.787 [Forwarding newSession on session null to remote] DEBUG io.netty.util.NetUtilInitializations - Loopback
interface: lo (lo, 0:0:0:0:0:0:0:1%lo)
05:43:58.788 [Forwarding newSession on session null to remote] DEBUG io.netty.util.NetUtil - /proc/sys/net/core/somaxconn:
4096
05:43:58.789 [Forwarding newSession on session null to remote] DEBUG io.netty.channel.DefaultChannelId - -
Dio.netty.machineId: 02:42:20:ff:fe:b5:c4:b5 (auto-detected)
05:43:58.842 [AsyncHttpClient-1-2] DEBUG io.netty.buffer.AbstractByteBuf - -Dio.netty.buffer.checkAccessible: true
05:43:58.842 [AsyncHttpClient-1-2] DEBUG io.netty.buffer.AbstractByteBuf - -Dio.netty.buffer.checkBounds: true
05:43:58.842 [AsyncHttpClient-1-2] DEBUG io.netty.util.ResourceLeakDetectorFactory - Loaded default ResourceLeakDetector:
io.netty.util.ResourceLeakDetector@37bf5376
05:43:58.854 [AsyncHttpClient-1-2] DEBUG org.asynchttpclient.netty.channel.NettyConnectListener - Using new Channel '[id:
0x00ea1271, L:/127.0.0.1:54194 - R:localhost/127.0.0.1:10156]' for 'POST' to '/session'
05:43:58.880 [AsyncHttpClient-1-2] DEBUG io.netty.util.Recycler - -Dio.netty.recycler.maxCapacityPerThread: 4096
05:43:58.880 [AsyncHttpClient-1-2] DEBUG io.netty.util.Recycler - -Dio.netty.recycler.ratio: 8
Type here to search
ENG
IN
11:14
04-05-2023

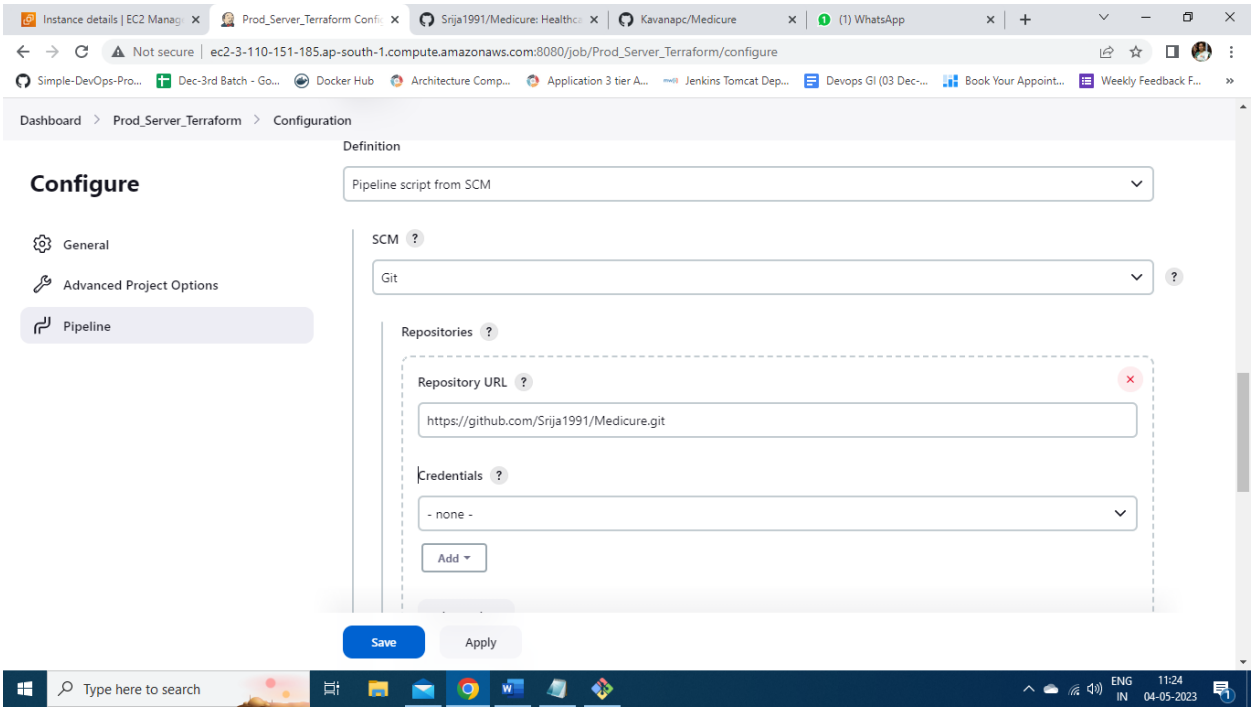


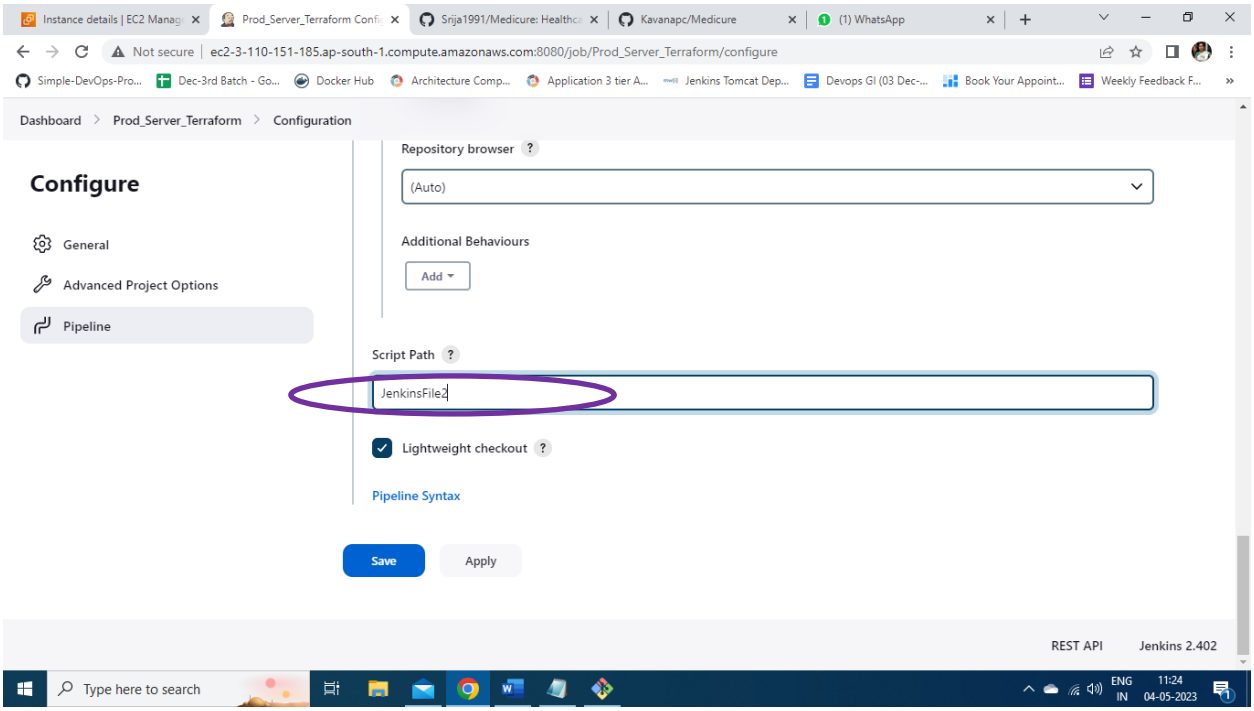
After successful execution of Selenium script, a new job named *Prod_server_Terrfaorm* is created to create Production server using terraform.



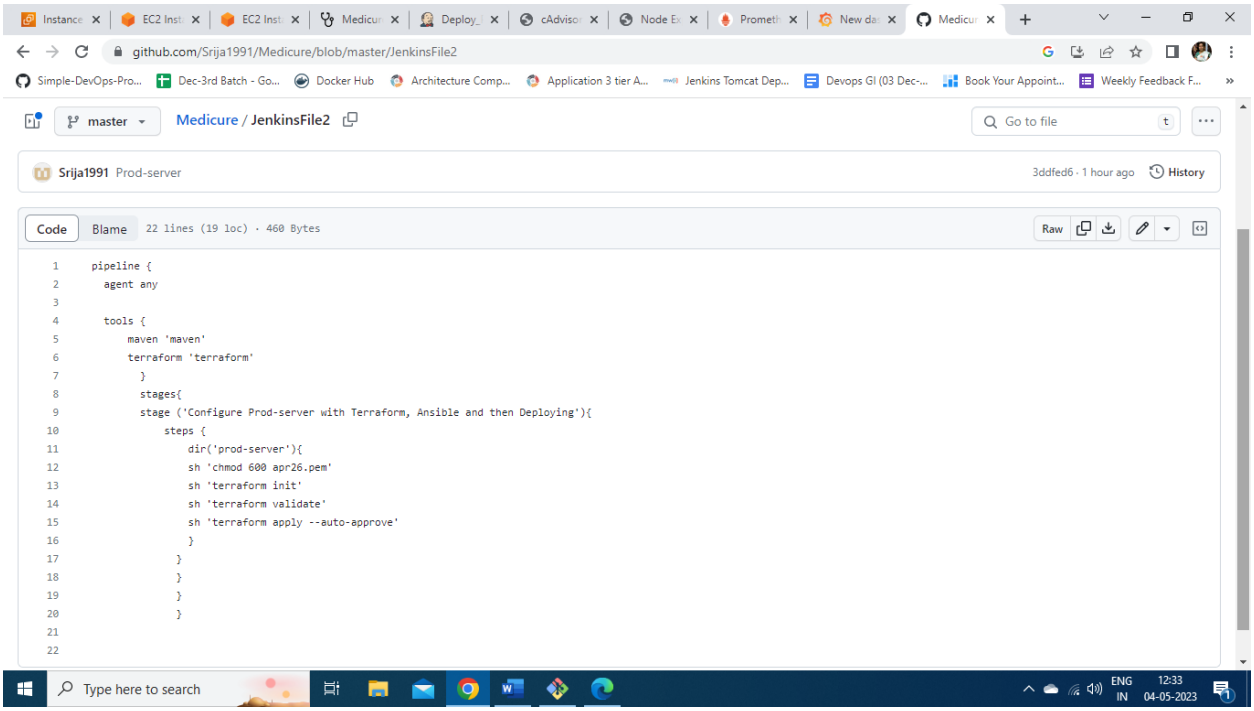


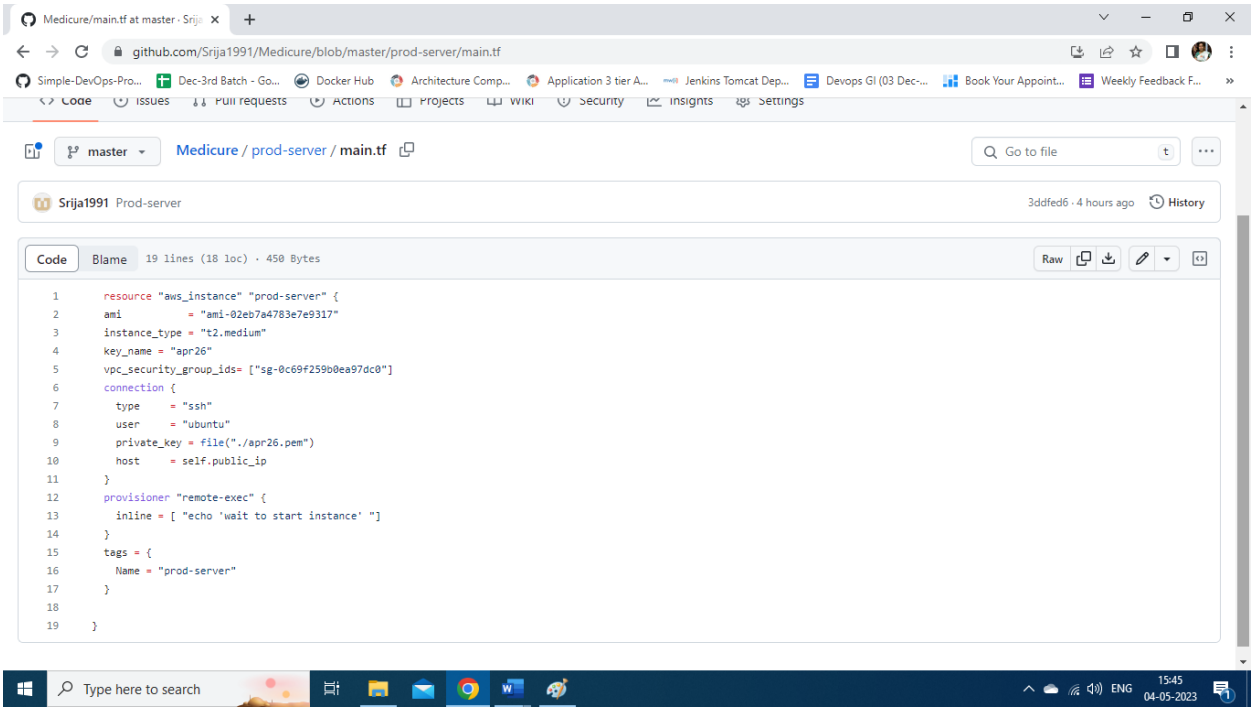
The pipeline script for this is provided through *JenkinsFile2* on the github.



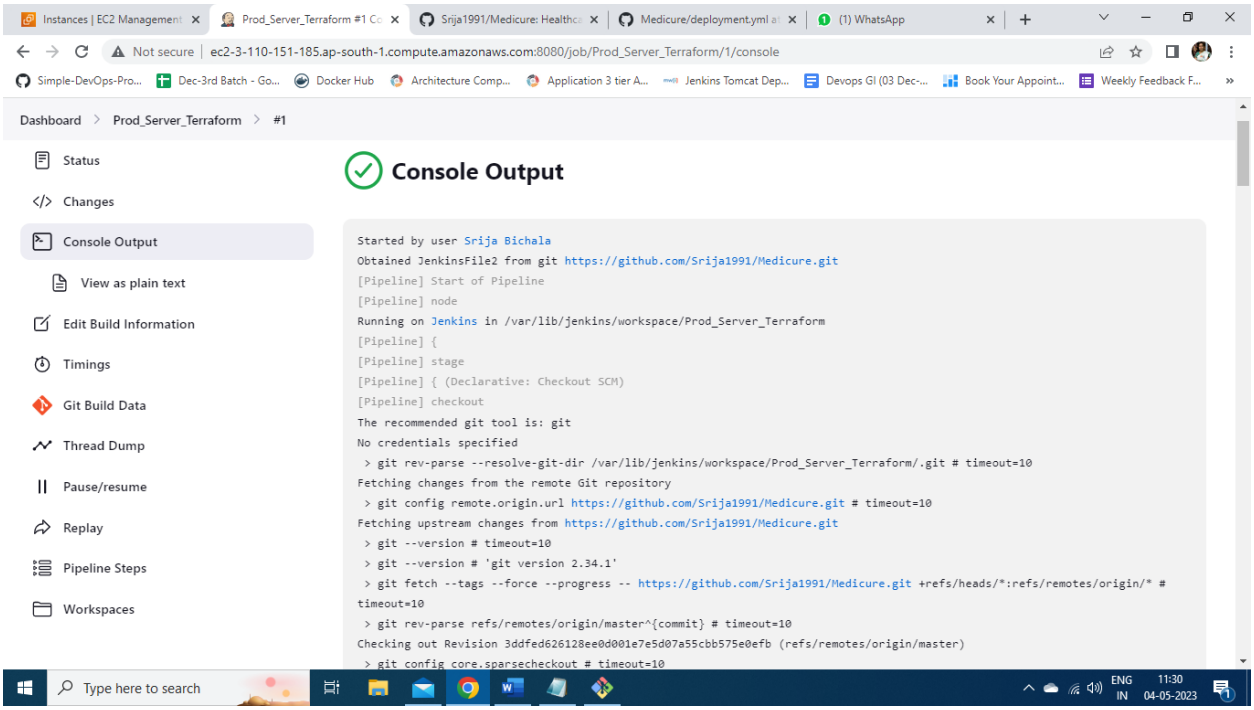


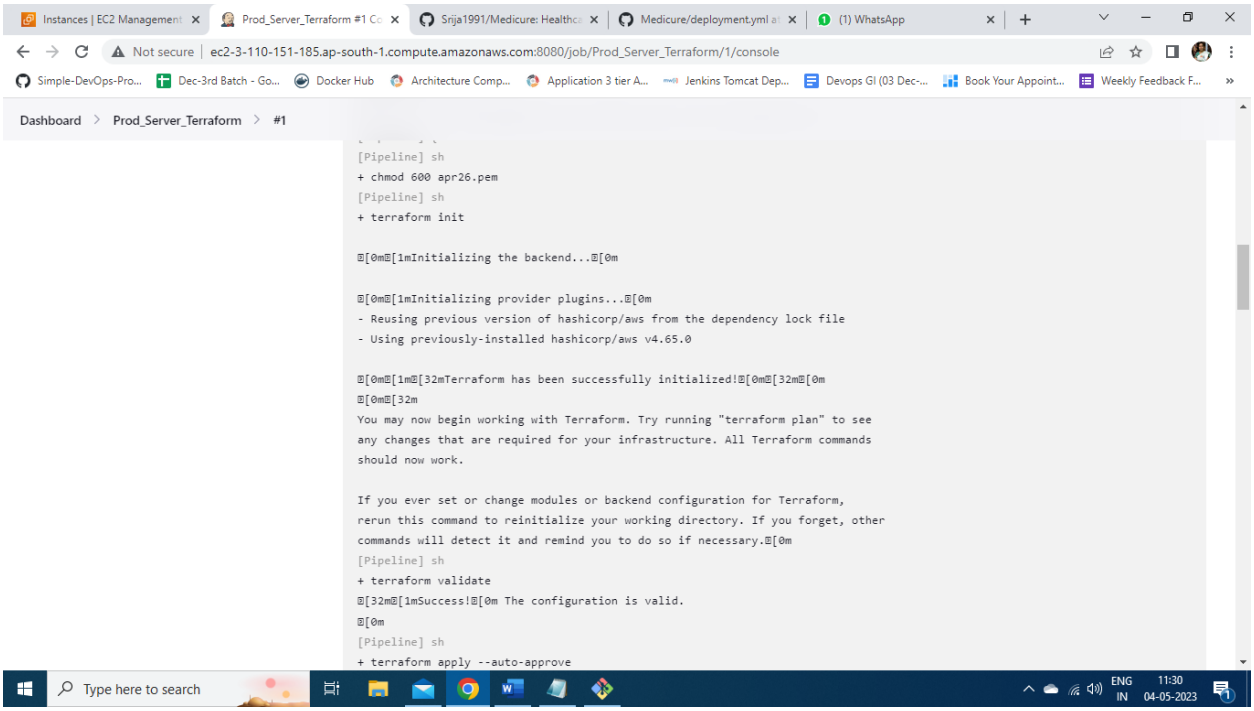
Here is the script to create new instance of t2.medium.

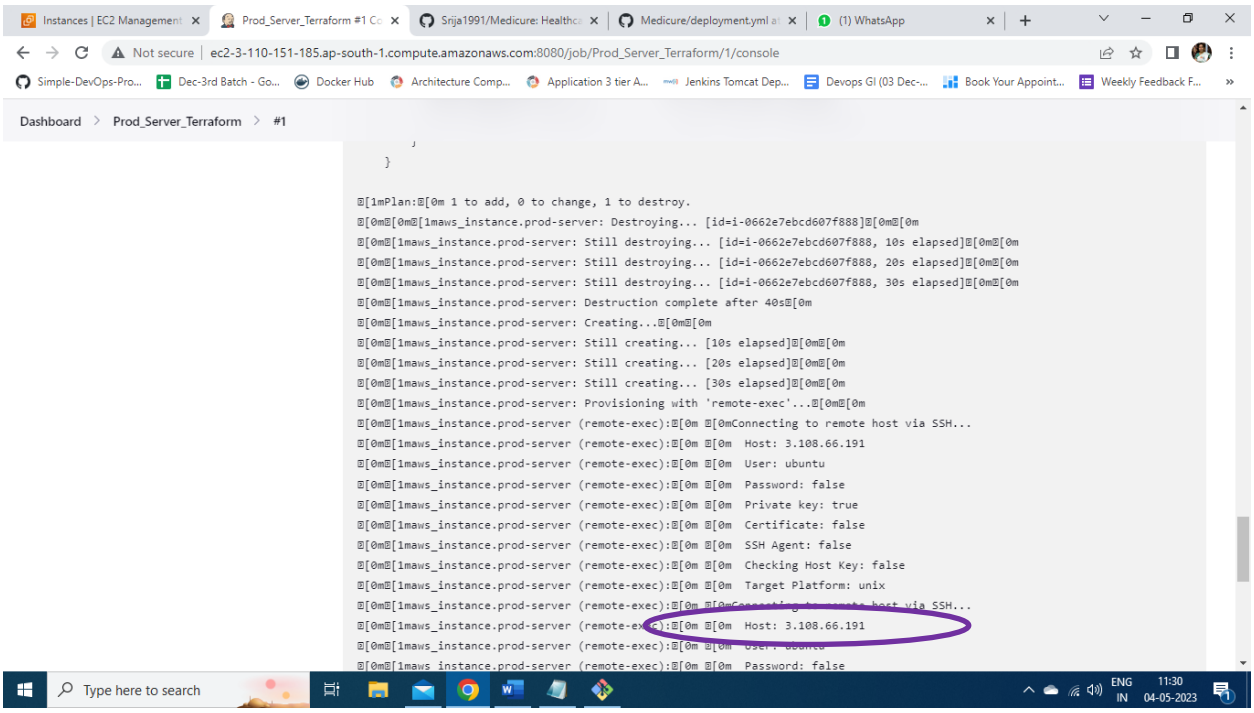


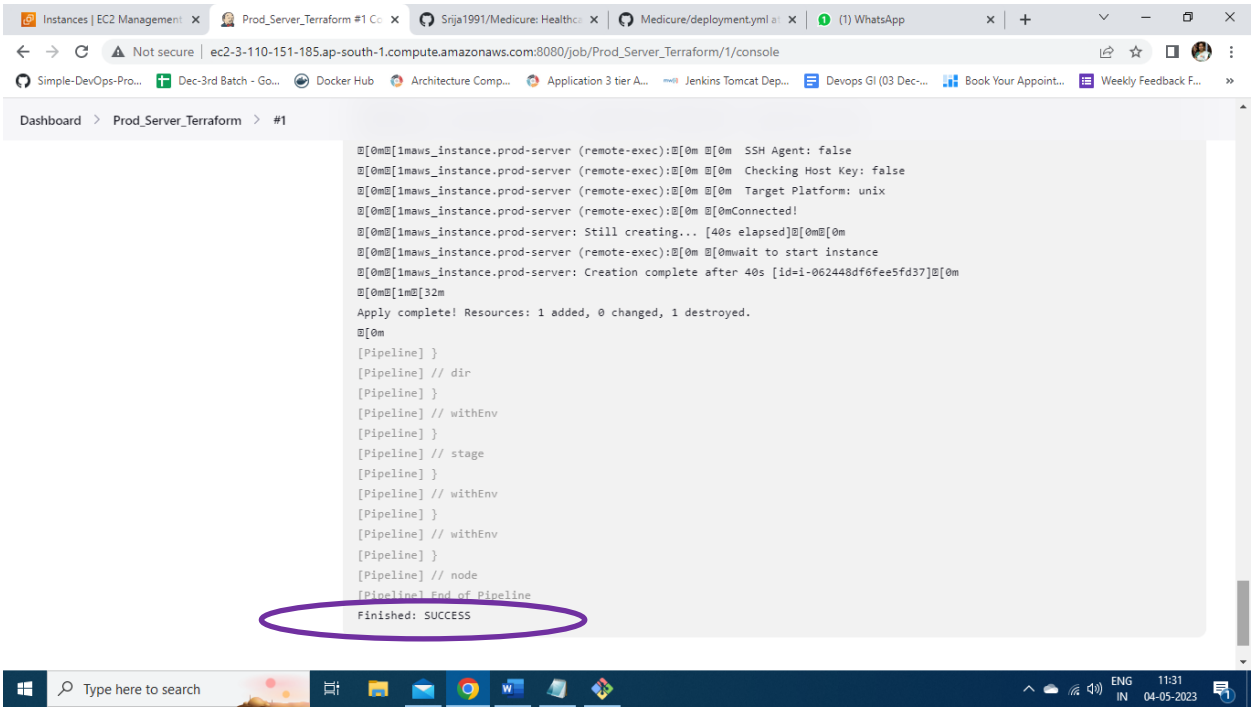


Now the job is executed and it is success creating a t2.medium instance which is manually configured for Kubernetes setup, since the project need to deploy using Kubernetes-cluster upon successful testing.

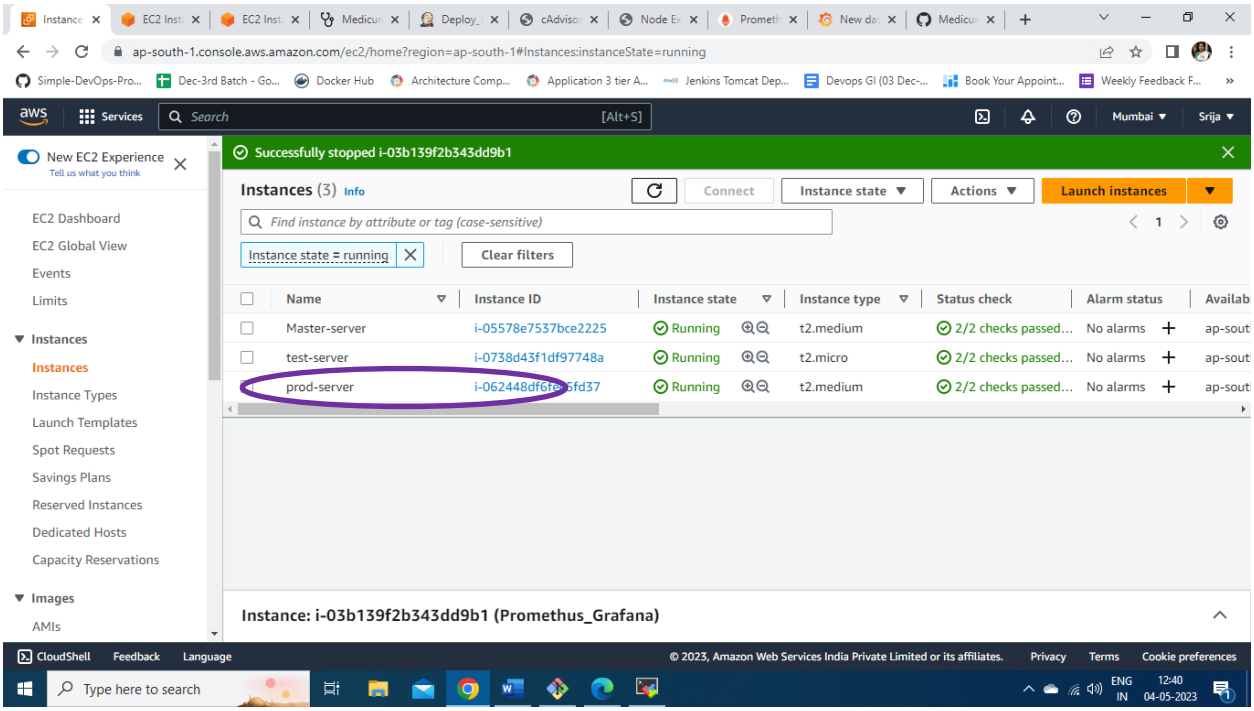


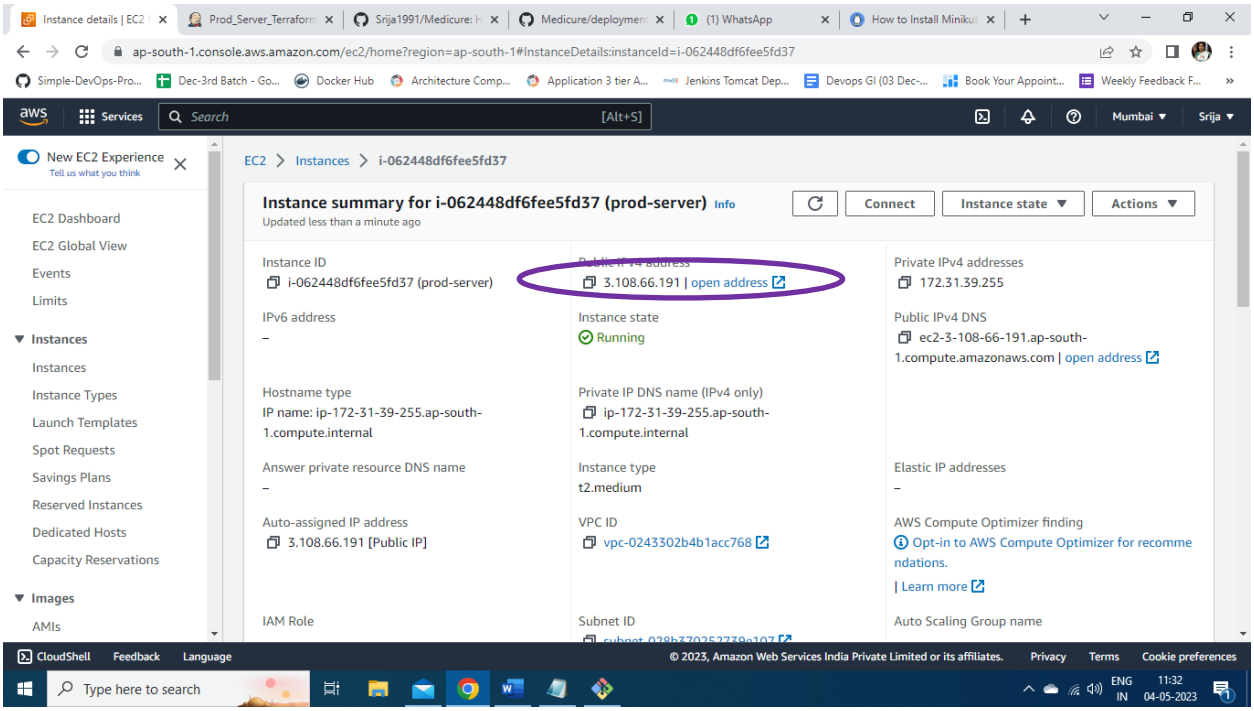

Instances | EC2 Management
Prod_Server_Terraform #1 Co
Srija1991/Medicure: Healthca
Medicure/deployment.yml at
(1) WhatsApp
Not secure | ec2-3-110-151-185.ap-south-1.compute.amazonaws.com:8080/job/Prod_Server_Terraform/1/console
Simple-DevOps-Pro...
Dec-3rd Batch - Go...
Docker Hub
Architecture Comp...
Application 3 tier A...
Jenkins Tomcat Dep...
Devops GI (03 Dec-...
Book Your Appoint...
Weekly Feedback F...
Dashboard > Prod_Server_Terraform > #1
[Pipeline] sh
+ chmod 600 apr26.pem
[Pipeline] sh
+ terraform init
[0m[1mInitializing the backend...[0m
[0m[1mInitializing provider plugins...[0m
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v4.65.0
[0m[1m[32mTerraform has been successfully initialized![0m[32m[0m
[0m[32m
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.[0m
[Pipeline] sh
+ terraform validate
[32m[1mSuccess![0m The configuration is valid.
[0m
[Pipeline] sh
+ terraform apply --auto-approve
Type here to search
ENG
IN
11:30
04-05-2023


Instances | EC2 Management
Prod_Server_Terraform #1 Co
Srija1991/Medicure: Healthca
Medicure/deployment.yml at
(1) WhatsApp
Not secure | ec2-3-110-151-185.ap-south-1.compute.amazonaws.com:8080/job/Prod_Server_Terraform/1/console
Simple-DevOps-Pro...
Dec-3rd Batch - Go...
Docker Hub
Architecture Comp...
Application 3 tier A...
Jenkins Tomcat Dep...
Devops GI (03 Dec-...
Book Your Appoint...
Weekly Feedback F...
Dashboard > Prod_Server_Terraform > #1
}
[1mPlan:[0m 1 to add, 0 to change, 1 to destroy.
[0m[0m[1maws_instance.prod-server: Destroying... [id=i-0662e7ebcd607f888][0m[0m
[0m[1maws_instance.prod-server: Still destroying... [id=i-0662e7ebcd607f888, 10s elapsed][0m[0m
[0m[1maws_instance.prod-server: Still destroying... [id=i-0662e7ebcd607f888, 20s elapsed][0m[0m
[0m[1maws_instance.prod-server: Still destroying... [id=i-0662e7ebcd607f888, 30s elapsed][0m[0m
[0m[1maws_instance.prod-server: Destruction complete after 40s[0m
[0m[1maws_instance.prod-server: Creating...[0m[0m
[0m[1maws_instance.prod-server: Still creating... [10s elapsed][0m[0m
[0m[1maws_instance.prod-server: Still creating... [20s elapsed][0m[0m
[0m[1maws_instance.prod-server: Still creating... [30s elapsed][0m[0m
[0m[1maws_instance.prod-server: Provisioning with 'remote-exec'...[0m[0m
[0m[1maws_instance.prod-server (remote-exec):[0m [0mConnecting to remote host via SSH...
[0m[1maws_instance.prod-server (remote-exec):[0m [0m Host: 3.108.66.191
[0m[1maws_instance.prod-server (remote-exec):[0m [0m User: ubuntu
[0m[1maws_instance.prod-server (remote-exec):[0m [0m Password: false
[0m[1maws_instance.prod-server (remote-exec):[0m [0m Private key: true
[0m[1maws_instance.prod-server (remote-exec):[0m [0m Certificate: false
[0m[1maws_instance.prod-server (remote-exec):[0m [0m SSH Agent: false
[0m[1maws_instance.prod-server (remote-exec):[0m [0m Checking Host Key: false
[0m[1maws_instance.prod-server (remote-exec):[0m [0m Target Platform: unix
[0m[1maws_instance.prod-server (remote-exec):[0m [0m Host: 3.108.66.191
[0m[1maws_instance.prod-server (remote-exec):[0m [0m Password: false
Type here to search
ENG
IN
11:30
04-05-2023

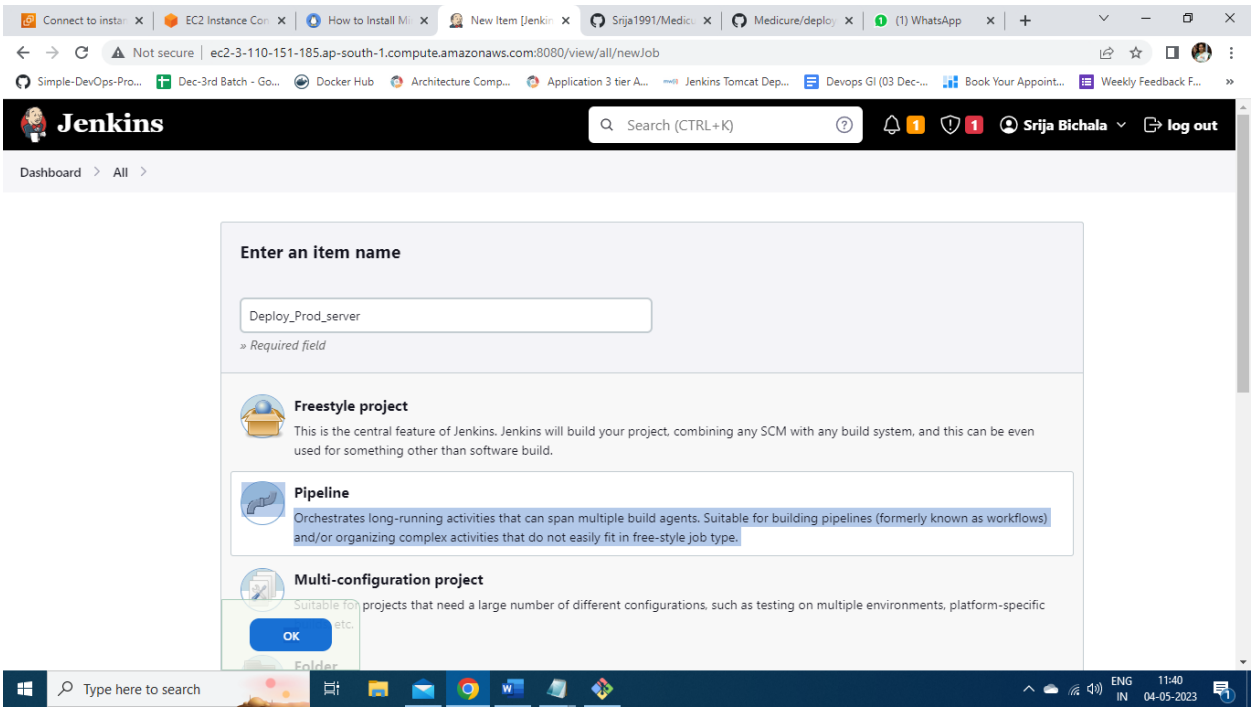


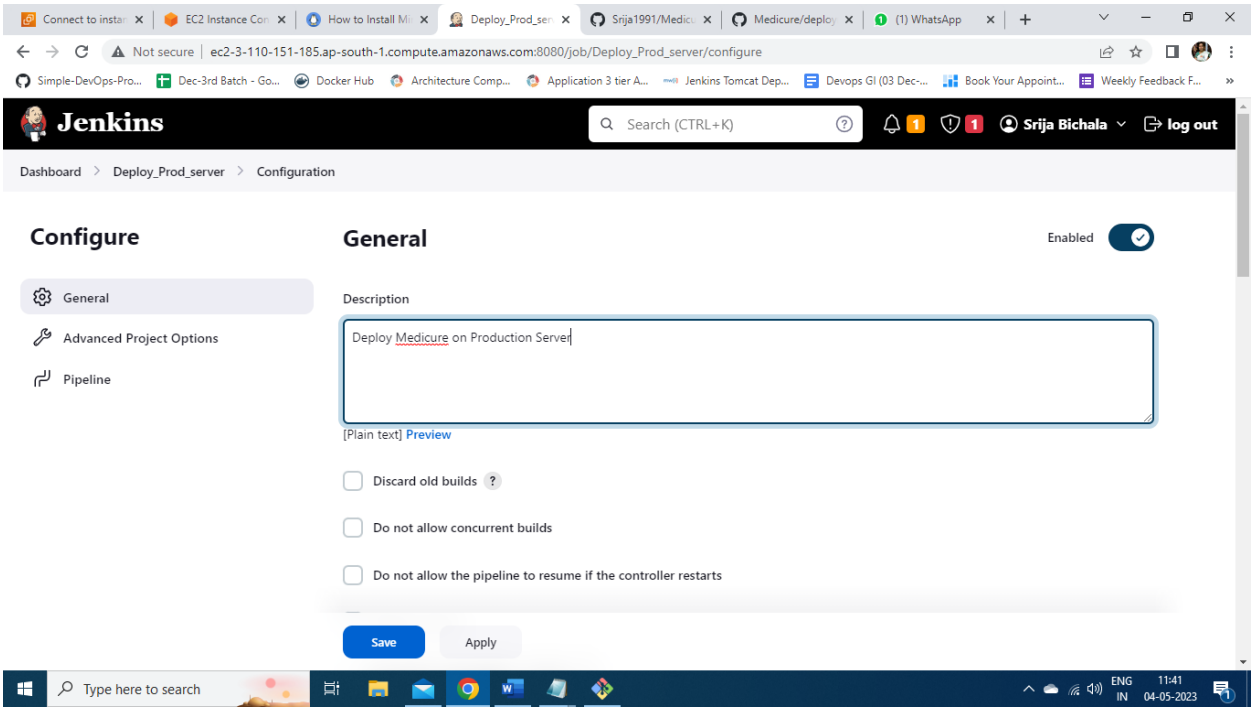
Here is the prod-server created and its specifications are shown here.





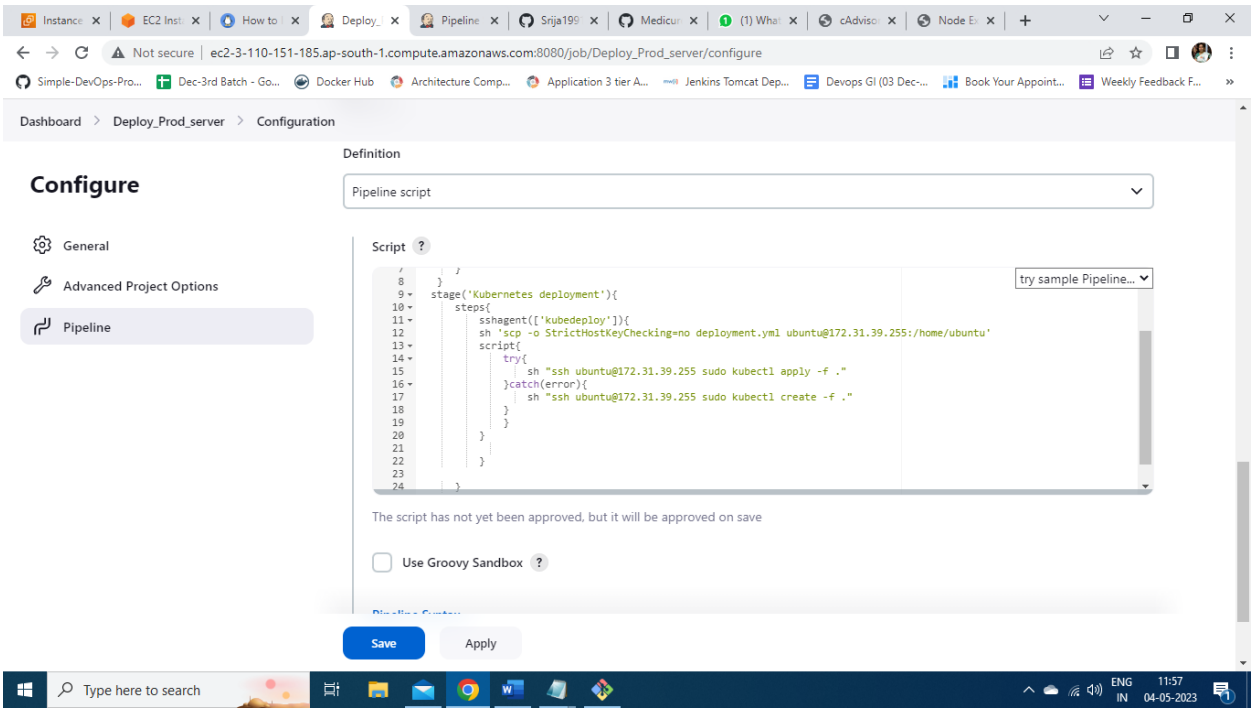
The prod-server is configured through Kubernetes setup and now to deploy the application through cluster a new job is created namely *Deploy_Prod_server.*



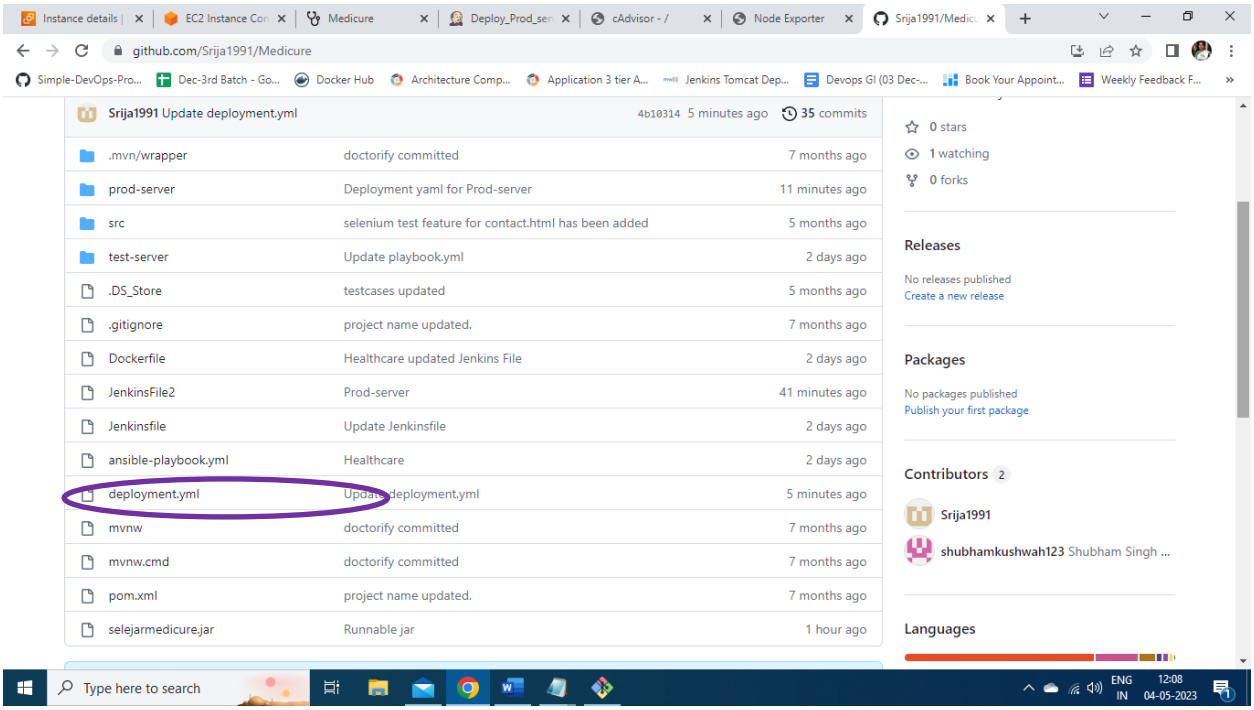


Here is the pipeline script to deploy through Kubernetes.

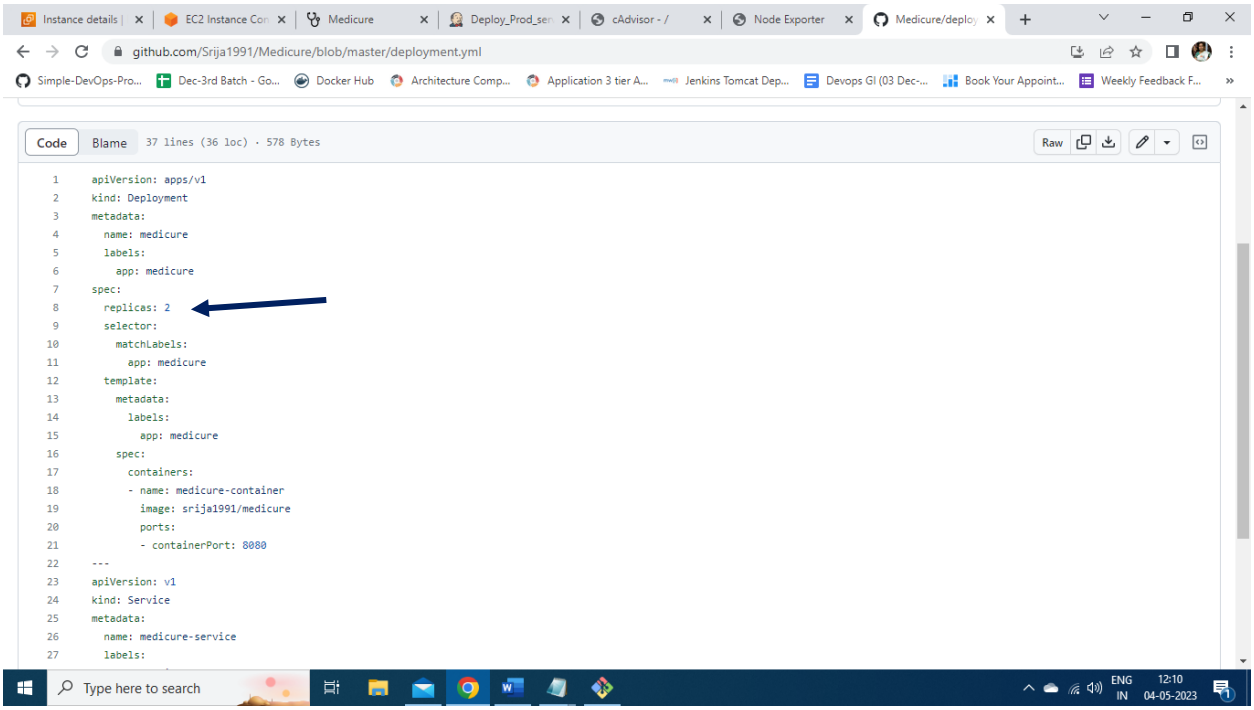
Stage : Create SSH connection and apply the deployment.yml file

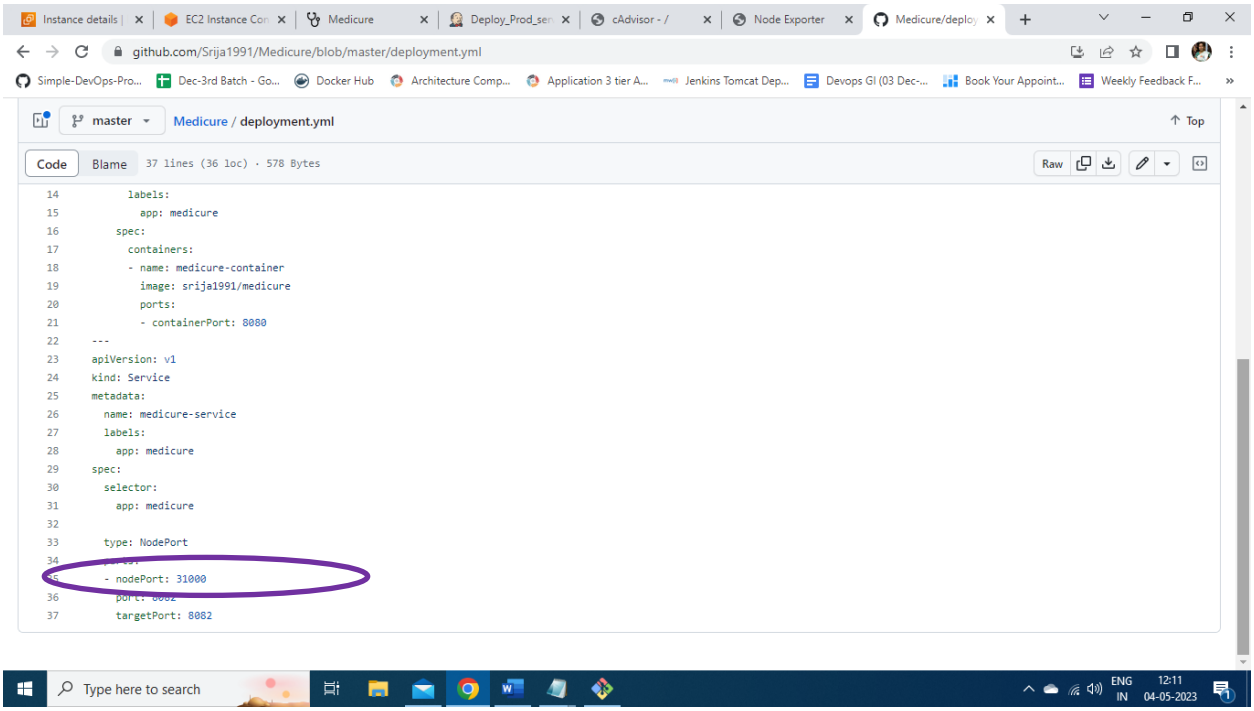


Here is the deployment.yml file pushed onto git repo.

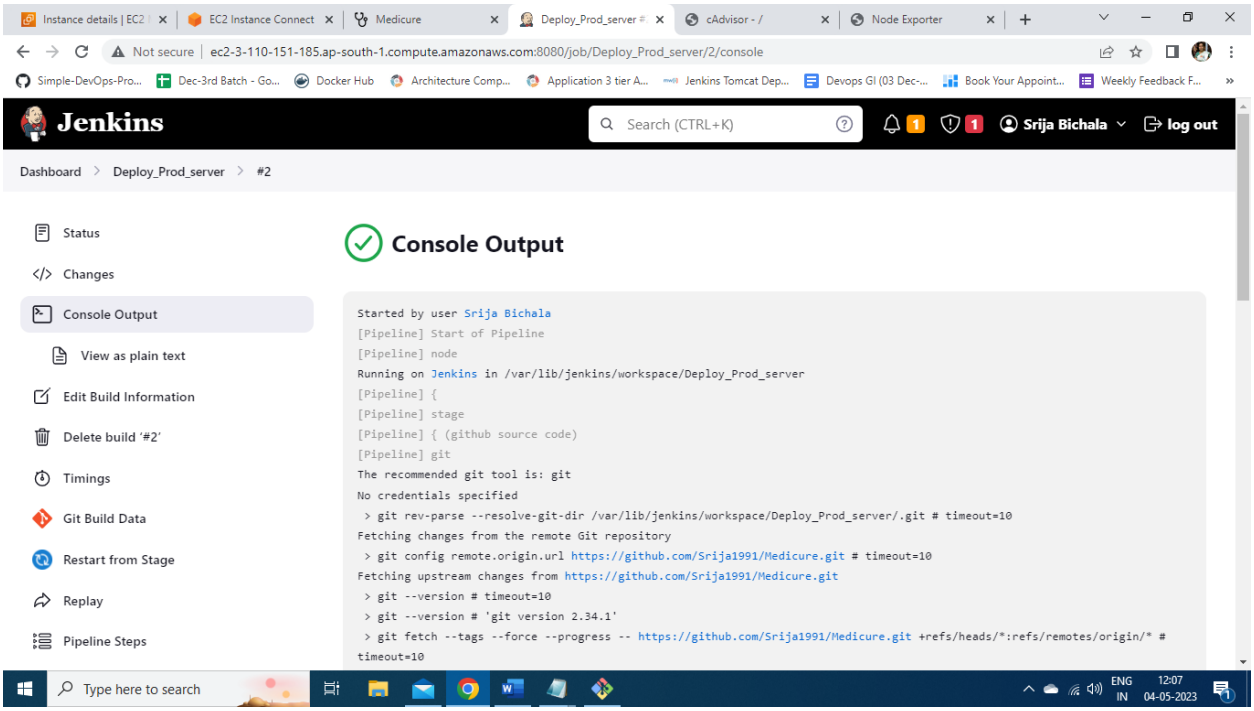


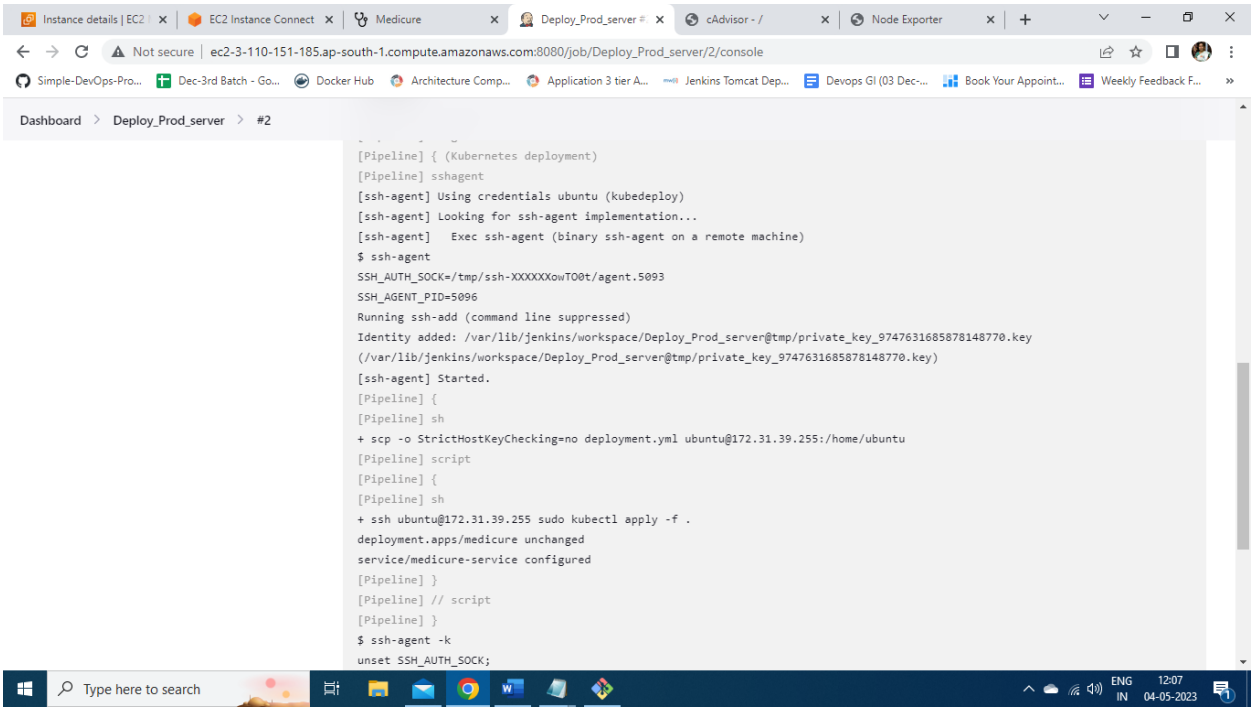
The deployment.yml file to create deployment using Kubernetes cluster with scaling factor 2, and using Nodeport service to deploy on port 31000.

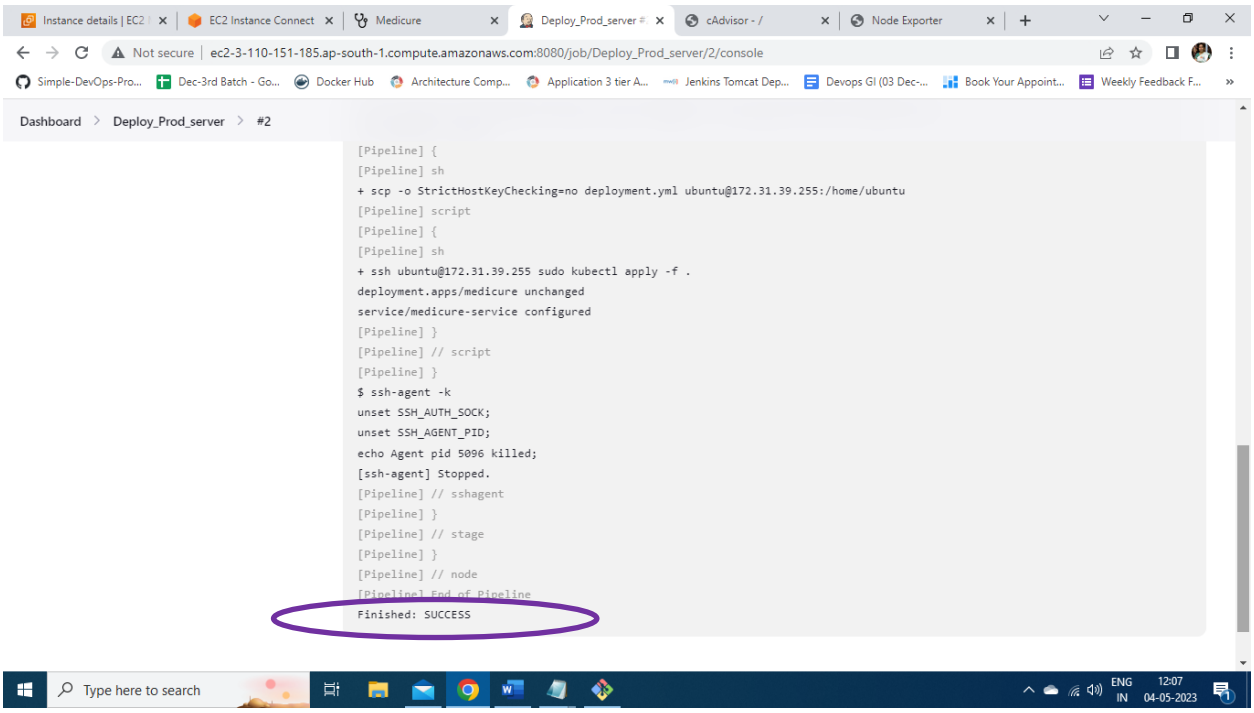




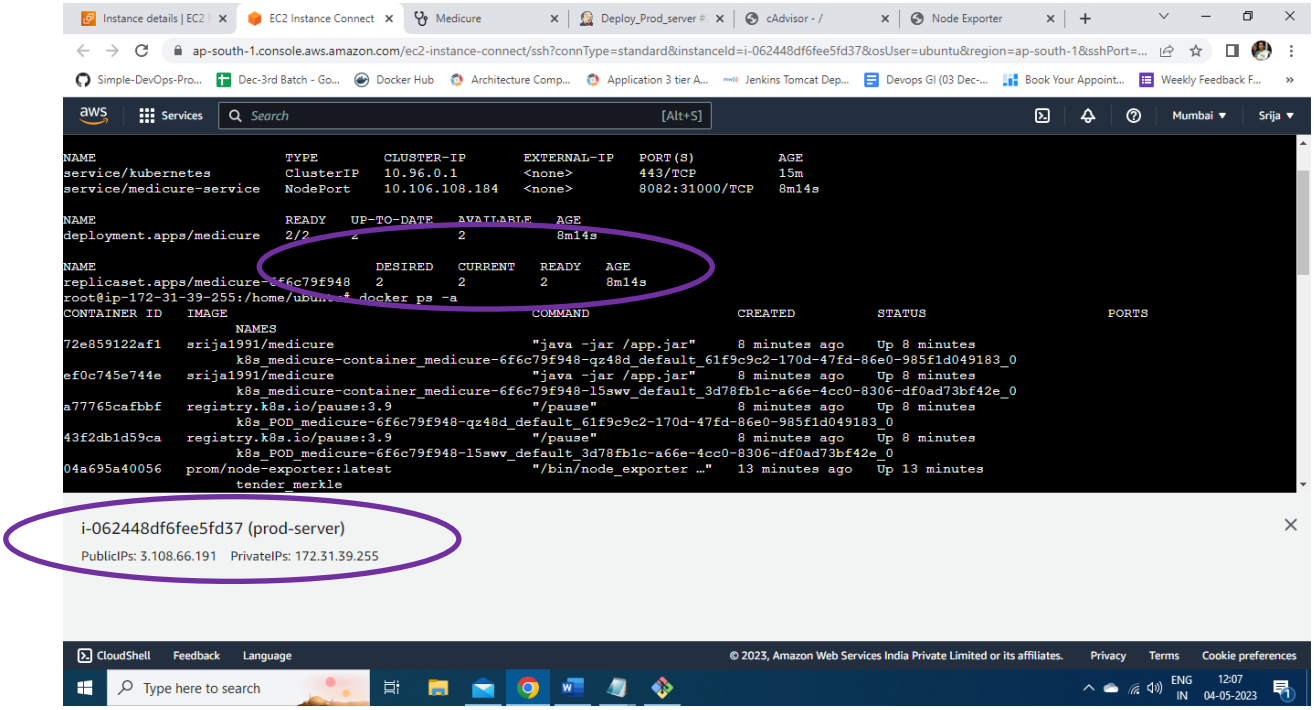
The job is run successfully and here is the console output .

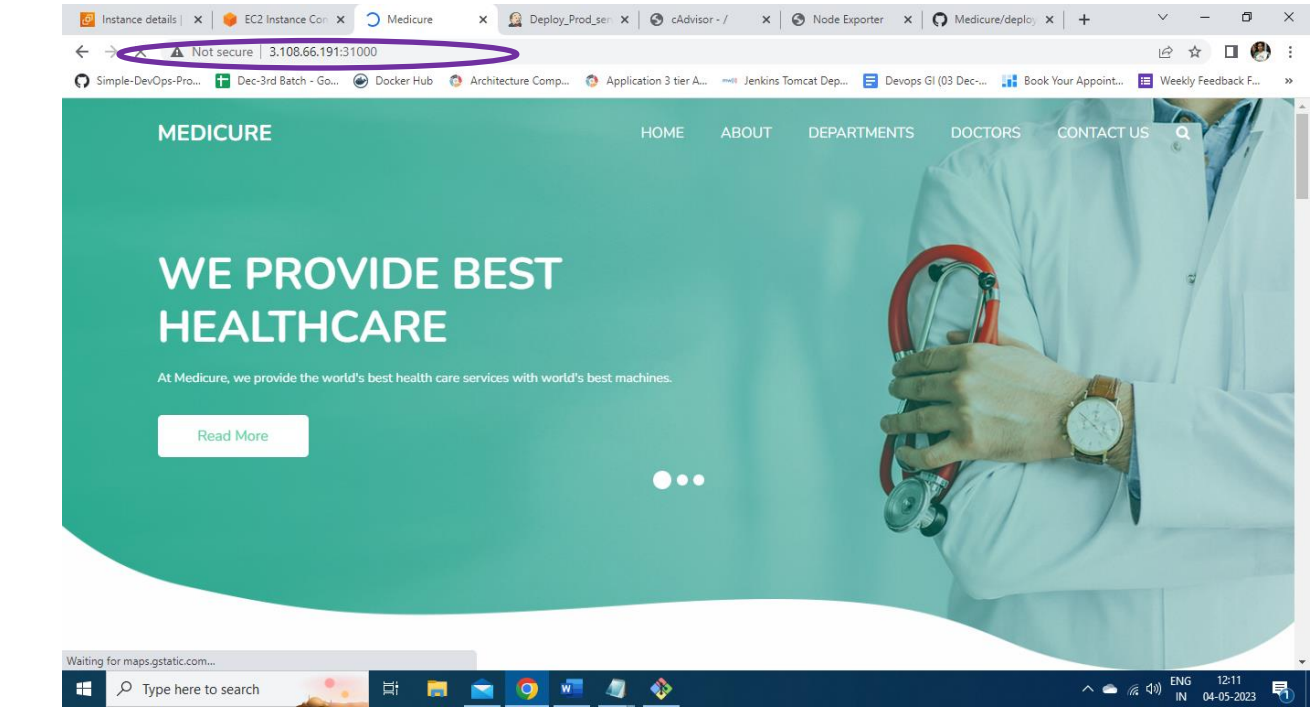



Instance details | EC2
EC2 Instance Connect
Medicure
Deploy_Prod_server #
cAdvisor - /
Node Exporter
Not secure | ec2-3-110-151-185.ap-south-1.compute.amazonaws.com:8080/job/Deploy_Prod_server/2/console
Simple-DevOps-Pro...
Dec-3rd Batch - Go...
Docker Hub
Architecture Comp...
Application 3 tier A...
Jenkins Tomcat Dep...
Devops GI (03 Dec-...
Book Your Appoint...
Weekly Feedback F...
Dashboard > Deploy_Prod_server > #2
[Pipeline] { (Kubernetes deployment)
[Pipeline] sshagent
[ssh-agent] Using credentials ubuntu (kubedeploy)
[ssh-agent] Looking for ssh-agent implementation...
[ssh-agent] Exec ssh-agent (binary ssh-agent on a remote machine)
$ ssh-agent
SSH_AUTH_SOCK=/tmp/ssh-XXXXXXowTO0t/agent.5093
SSH_AGENT_PID=5096
Running ssh-add (command line suppressed)
Identity added: /var/lib/jenkins/workspace/Deploy_Prod_server@tmp/private_key_9747631685878148770.key
(/var/lib/jenkins/workspace/Deploy_Prod_server@tmp/private_key_9747631685878148770.key)
[ssh-agent] Started.
[Pipeline] {
[Pipeline] sh
+ scp -o StrictHostKeyChecking=no deployment.yml ubuntu@172.31.39.255:/home/ubuntu
[Pipeline] script
[Pipeline] {
[Pipeline] sh
+ ssh ubuntu@172.31.39.255 sudo kubectl apply -f .
deployment.apps/medicure unchanged
service/medicure-service configured
[Pipeline] }
[Pipeline] // script
[Pipeline] }
$ ssh-agent -k
unset SSH_AUTH_SOCK;
Type here to search
ENG
IN
12:07
04-05-2023


Instance details | EC2
EC2 Instance Connect
Medicure
Deploy_Prod_server #
cAdvisor - /
Node Exporter
Not secure | ec2-3-110-151-185.ap-south-1.compute.amazonaws.com:8080/job/Deploy_Prod_server/2/console
Simple-DevOps-Pro...
Dec-3rd Batch - Go...
Docker Hub
Architecture Comp...
Application 3 tier A...
Jenkins Tomcat Dep...
Devops GI (03 Dec-...
Book Your Appoint...
Weekly Feedback F...
Dashboard > Deploy_Prod_server > #2
[Pipeline] {
[Pipeline] sh
+ scp -o StrictHostKeyChecking=no deployment.yml ubuntu@172.31.39.255:/home/ubuntu
[Pipeline] script
[Pipeline] {
[Pipeline] sh
+ ssh ubuntu@172.31.39.255 sudo kubectl apply -f .
deployment.apps/medicure unchanged
service/medicure-service configured
[Pipeline] }
[Pipeline] // script
[Pipeline] }
$ ssh-agent -k
unset SSH_AUTH_SOCK;
unset SSH_AGENT_PID;
echo Agent pid 5096 killed;
[ssh-agent] Stopped.
[Pipeline] // sshagent
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
Type here to search
ENG
IN
12:07
04-05-2023

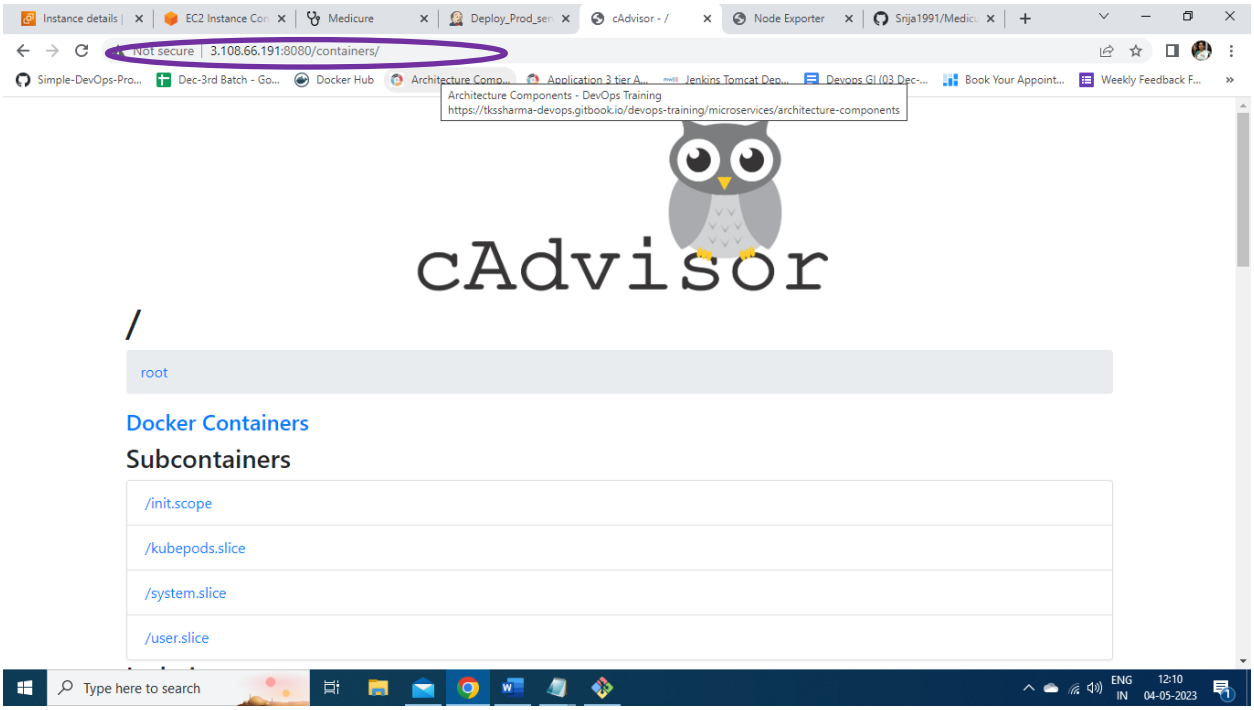
The prod-server is manually entered and checked for the pods and deployments using command *kubectl get all* . Here we can see the deployments 2/2 running stating that two pods are created using cluster on the port 31000. IP address of prod_server is 3.108.66.191.



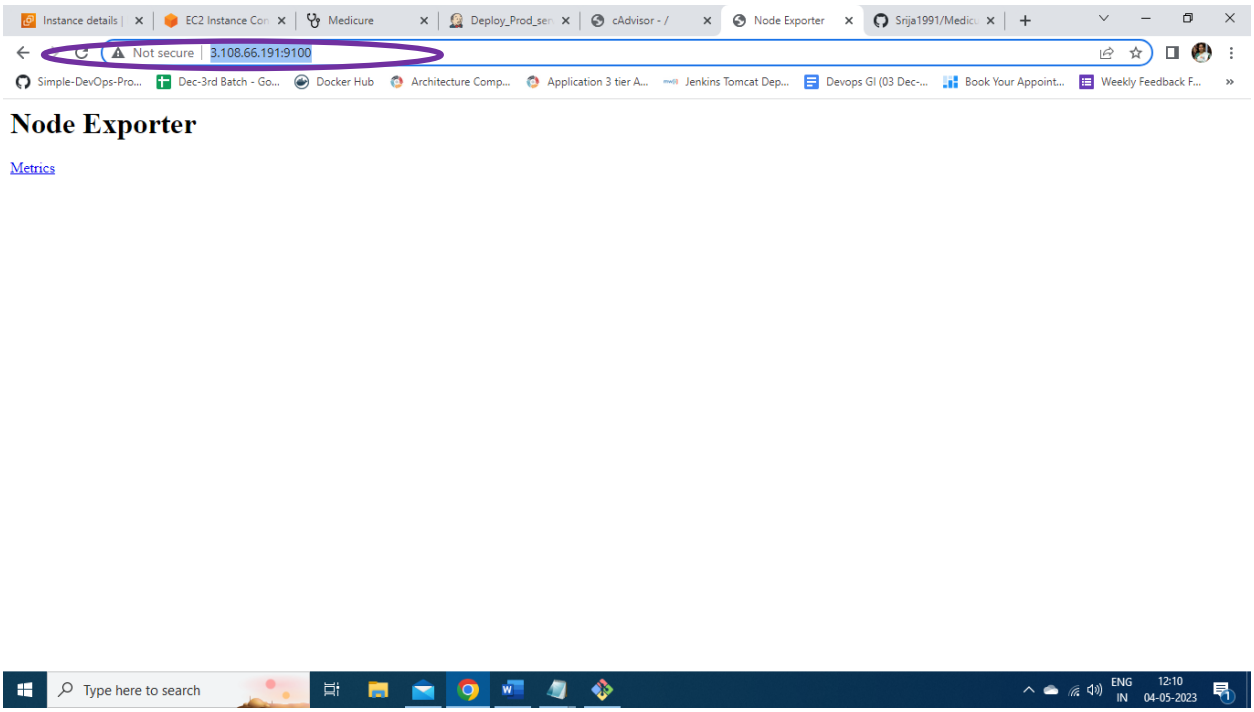


On the same production server cAdvisor and Node-exporter containers are started to take the metrics for monitoring using Prometheus and Grafana.

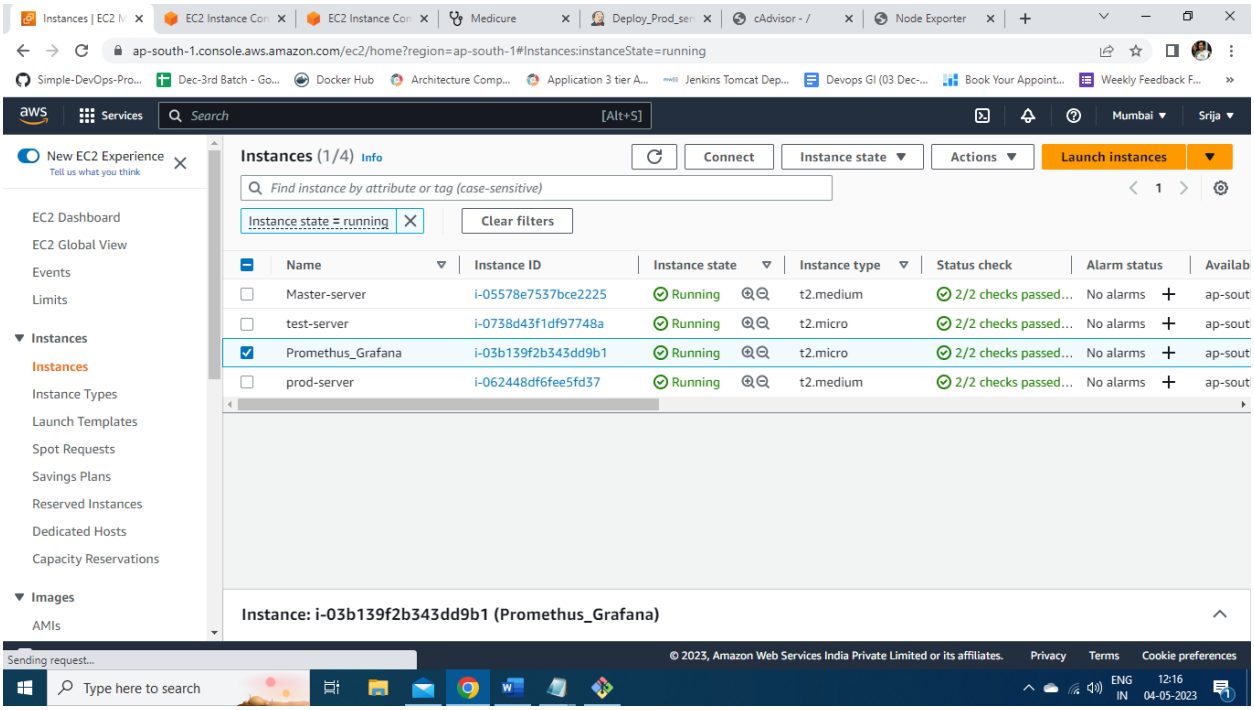
Here is the cAdvisor running on port 8080 on production server.



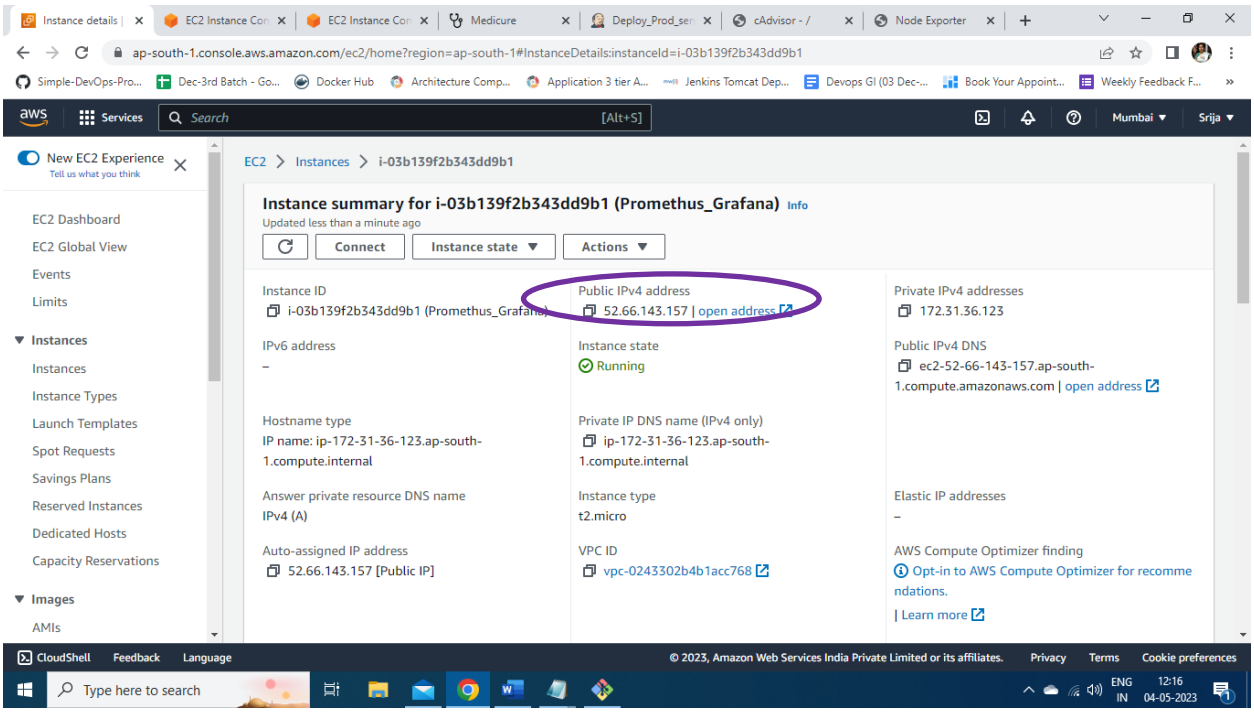
Here is the Node-exporter running on port 9100 on prod_server.



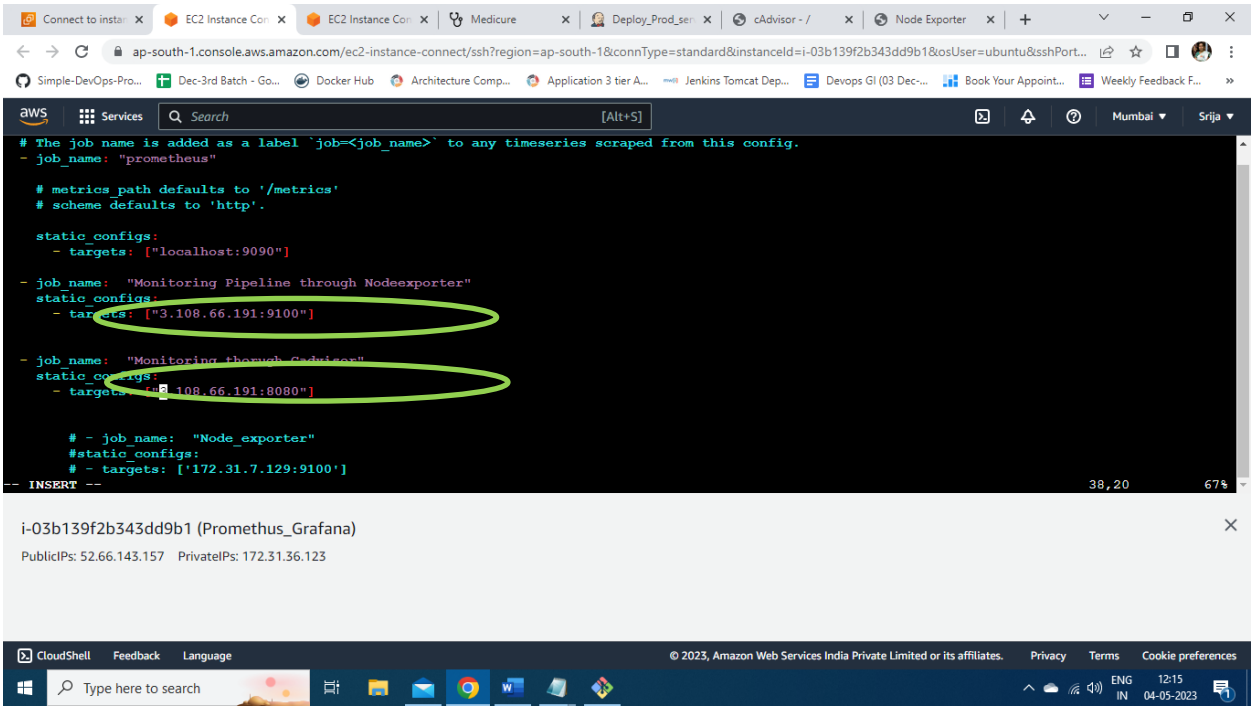
Now Prometheus_Grafana Server is configured to monitor the production server.



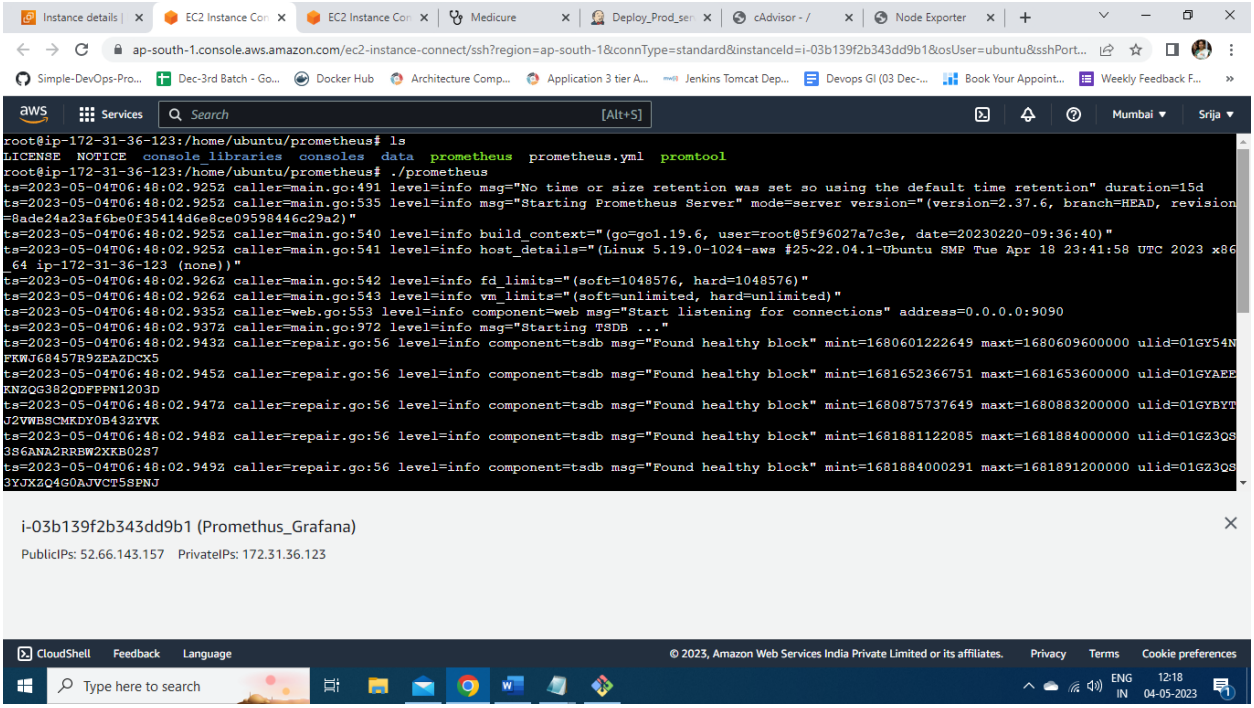
Here is the Instance summary of the monitoring server.



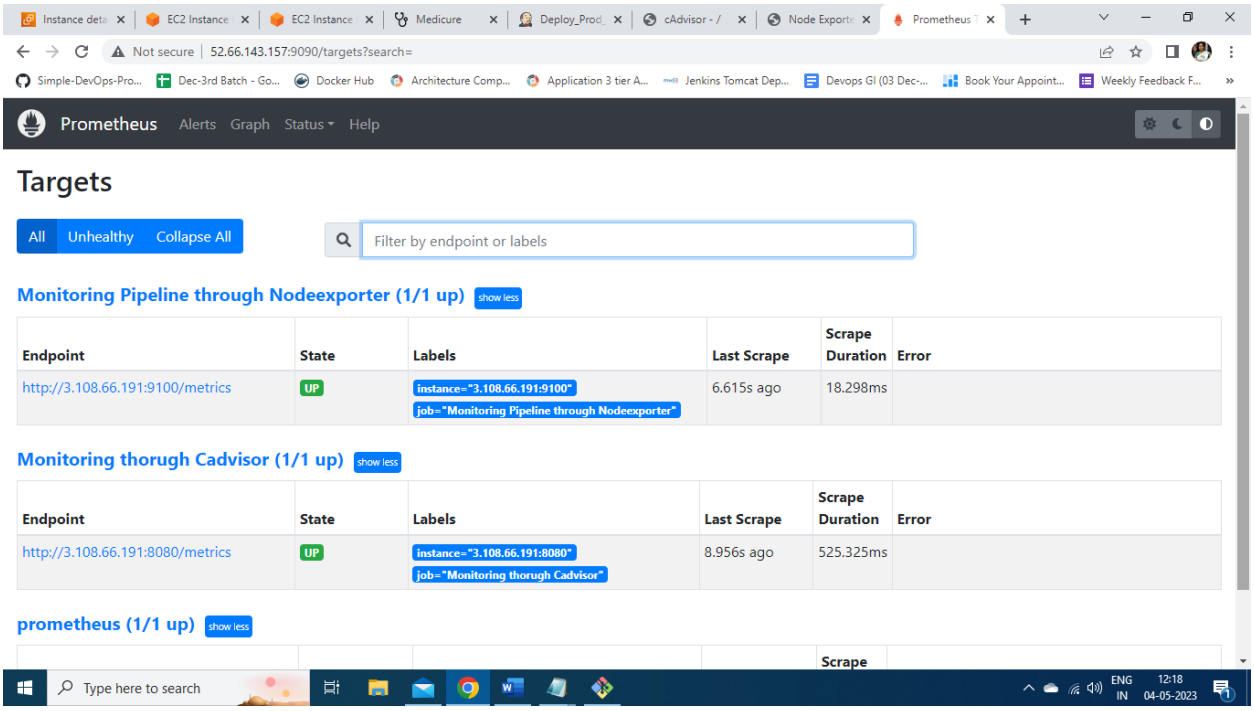
The monitoring server is configured by adding the IP address of cAdvisor and node-exporter available from production server in the *prometheus.yml* file.



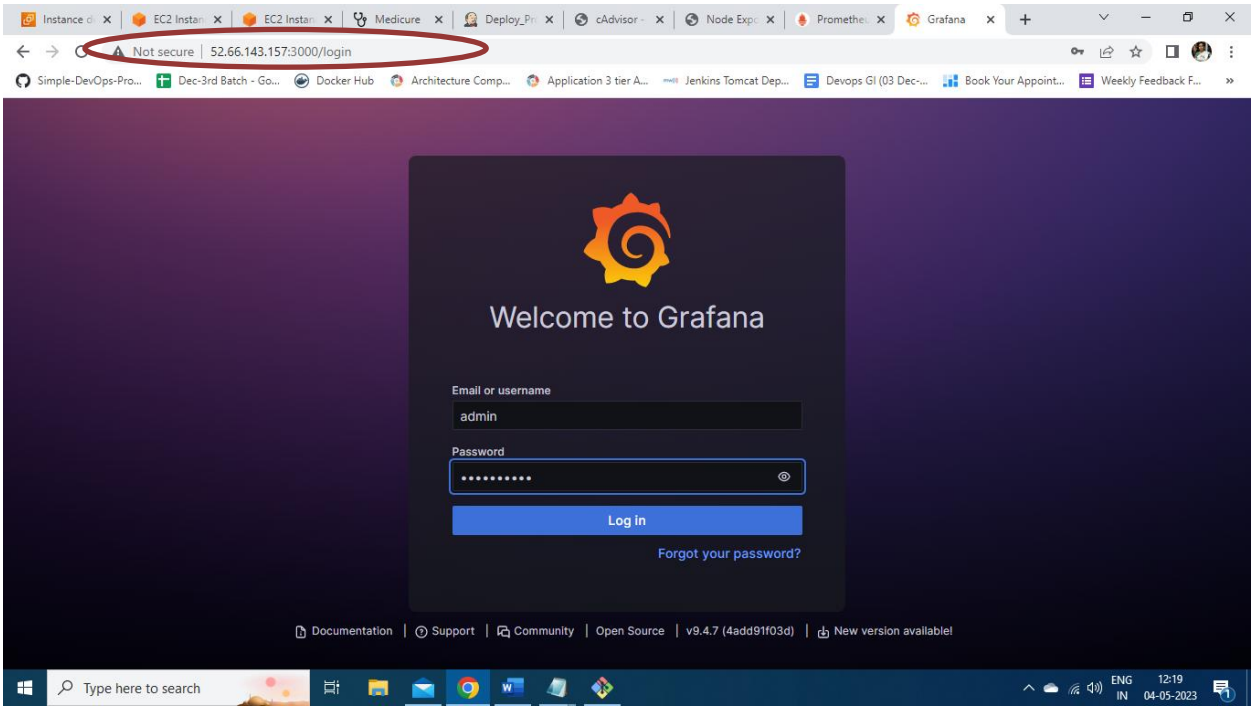
The Prometheus is started using command ./prometheus.



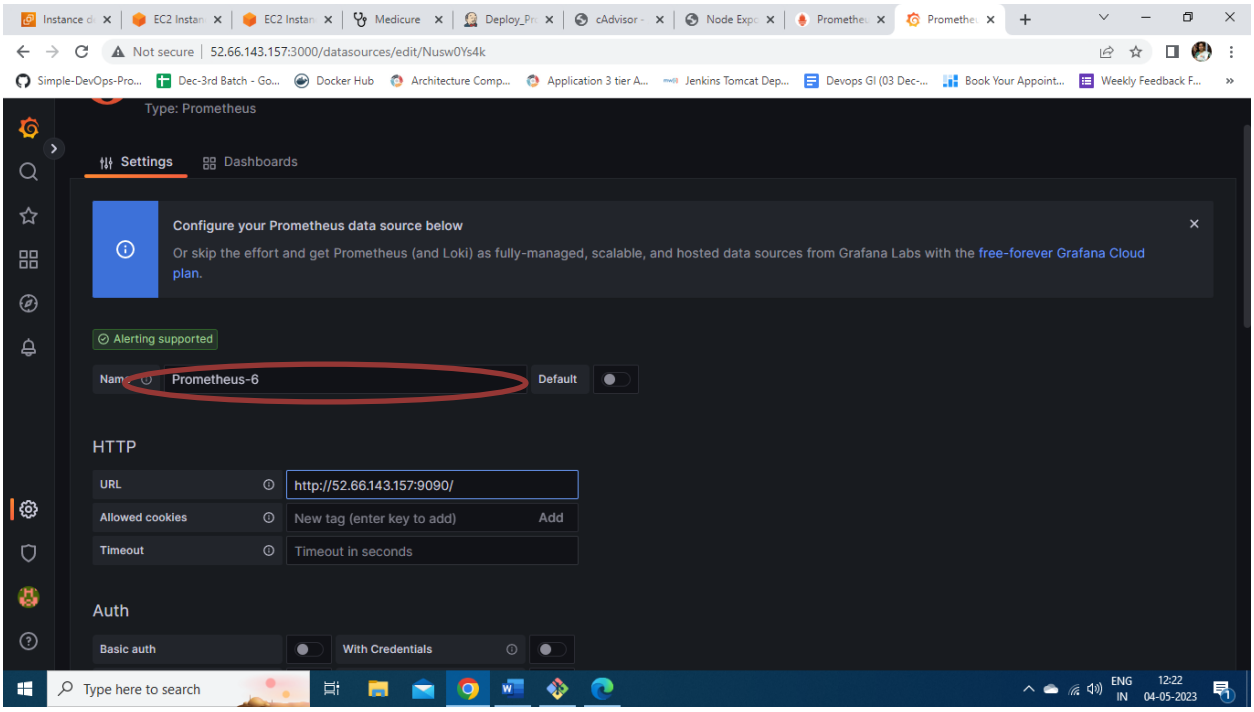
Here we can see the metrics are added as Prometheus targets on port number 9090 of monitoring server successfully and are available to monitor.

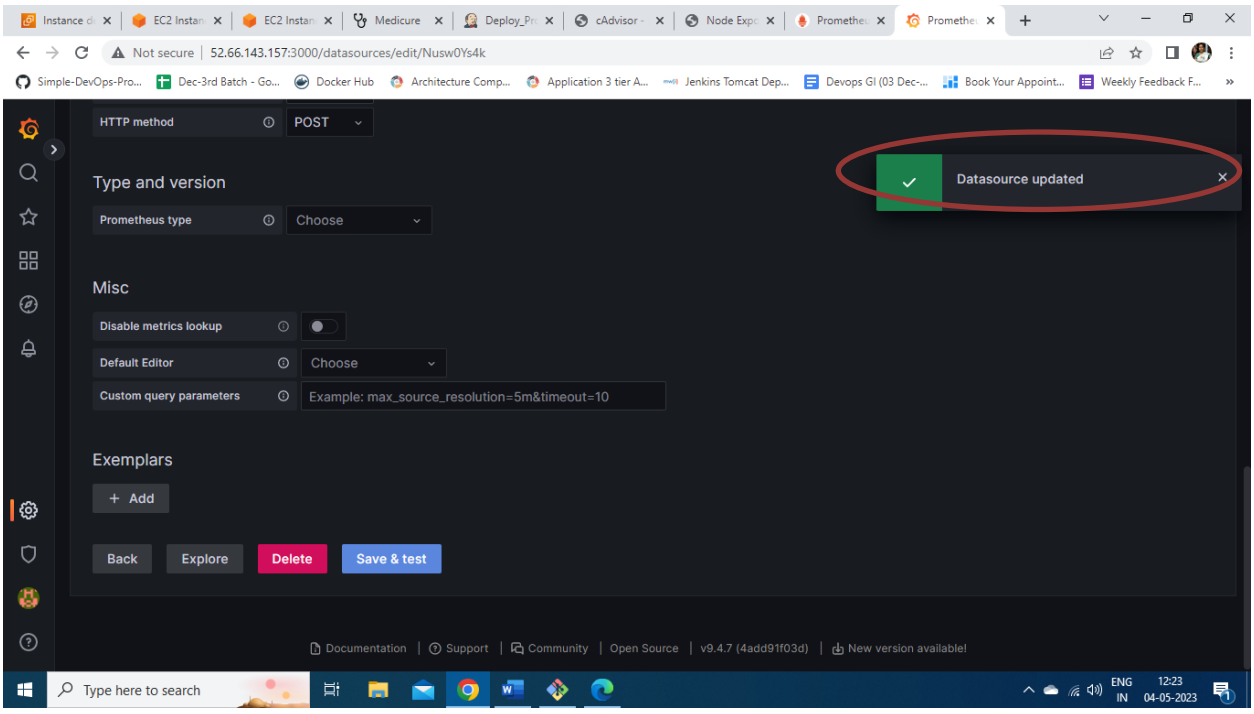


From the monitoring server port number 3000 is allocated to Grafana monitoring and logged in to monitor.

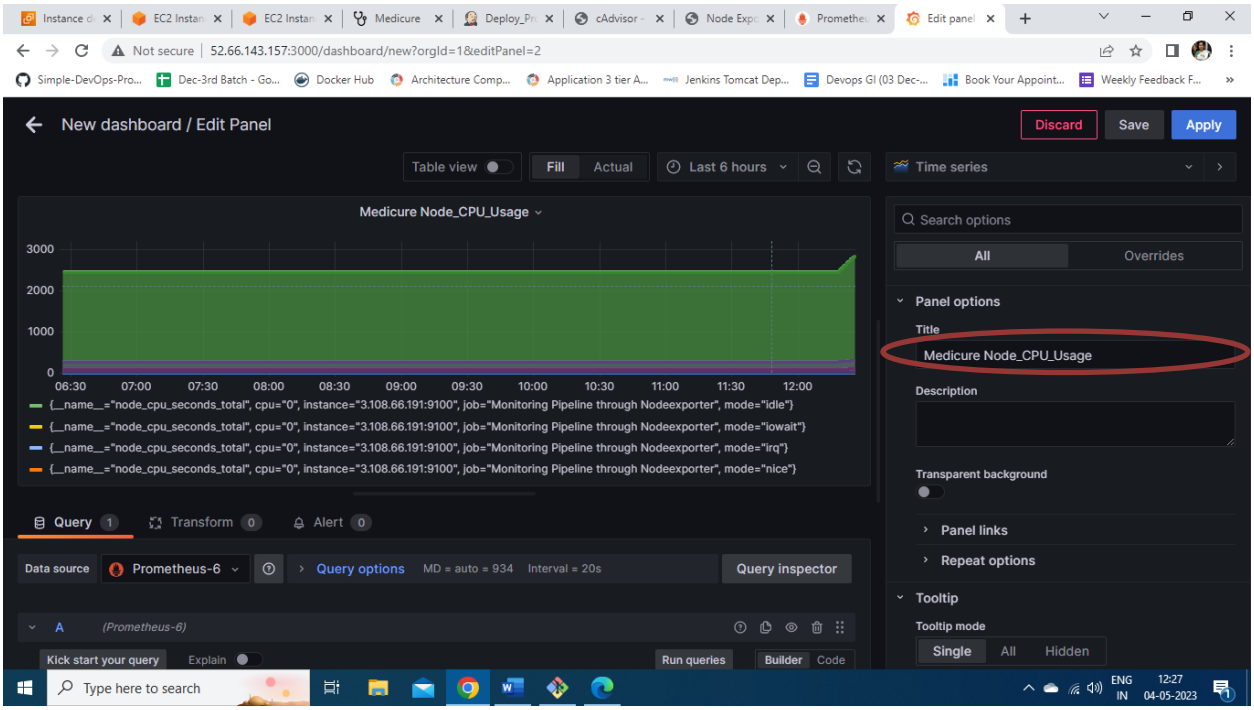


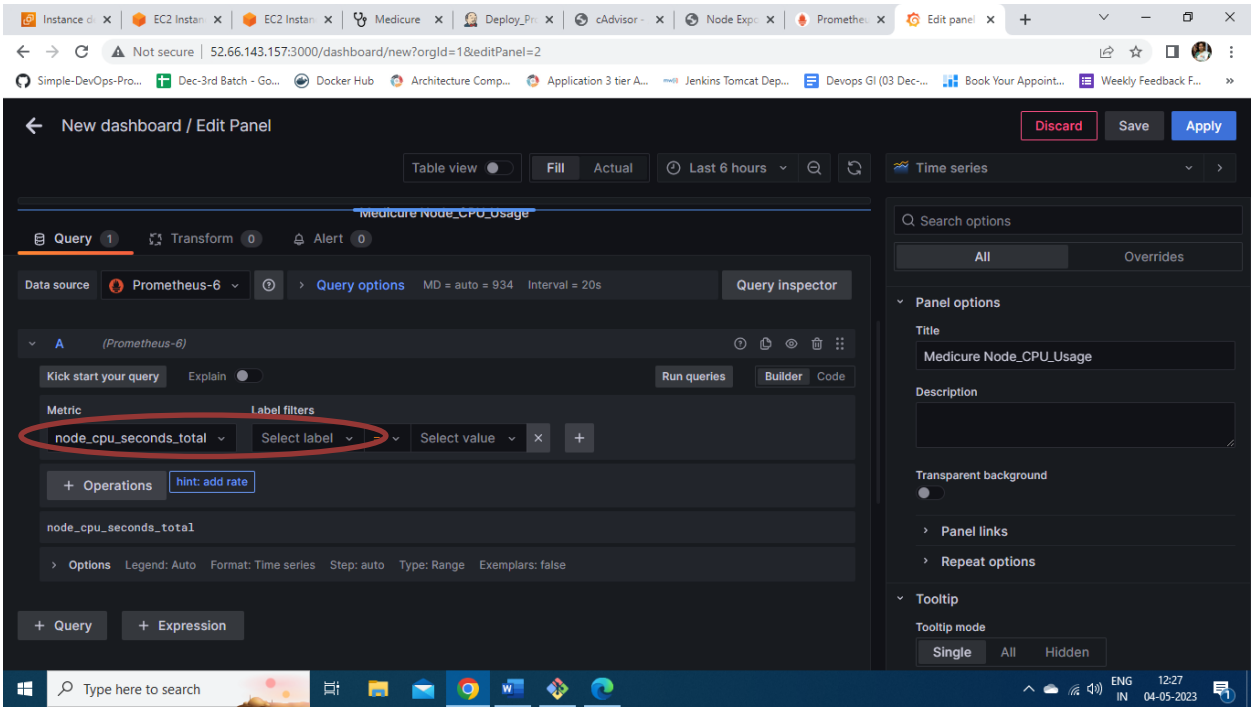
A new datasource is added named prometheus6 and configured to collect metrics.



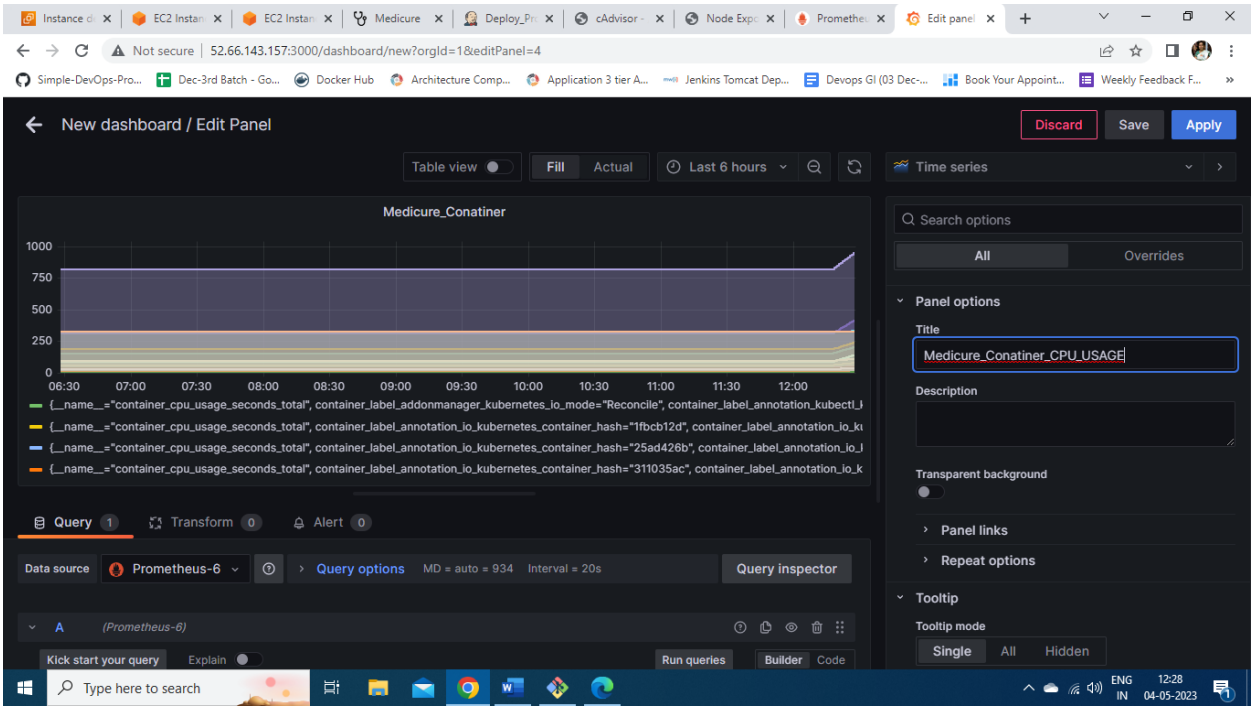


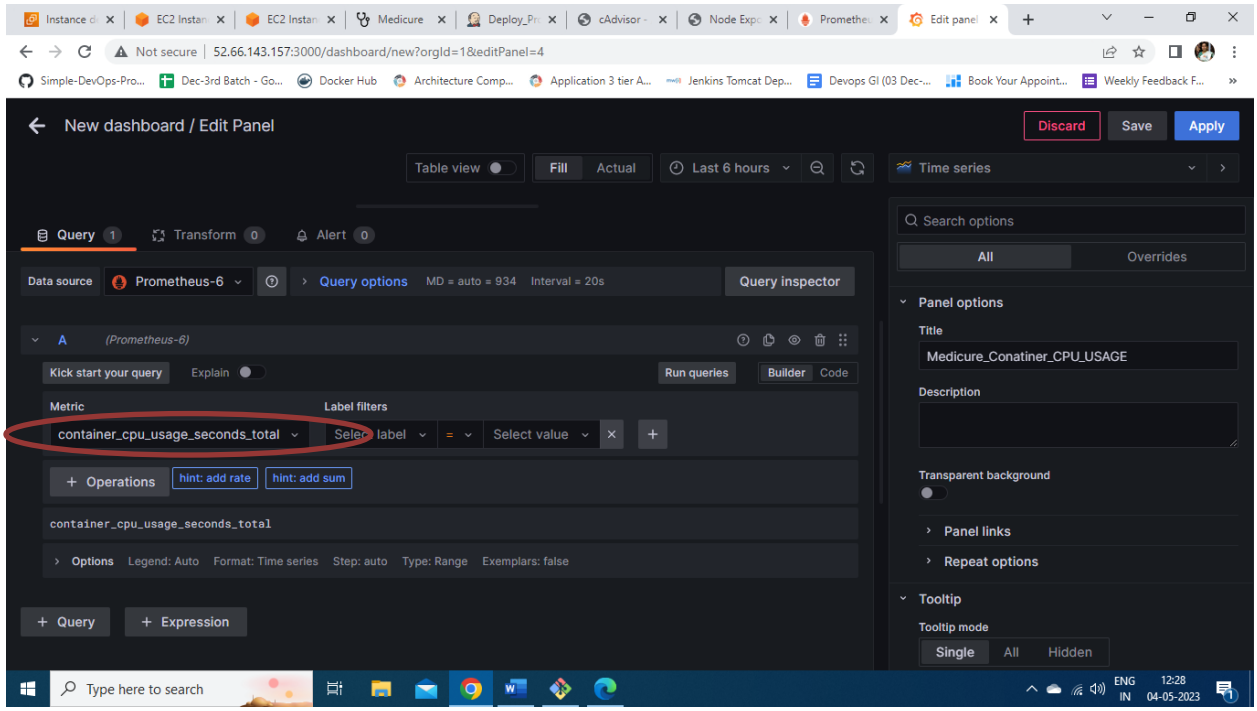
Now a Dashboard is created and a query called node_cpu_usage is run and monitored.



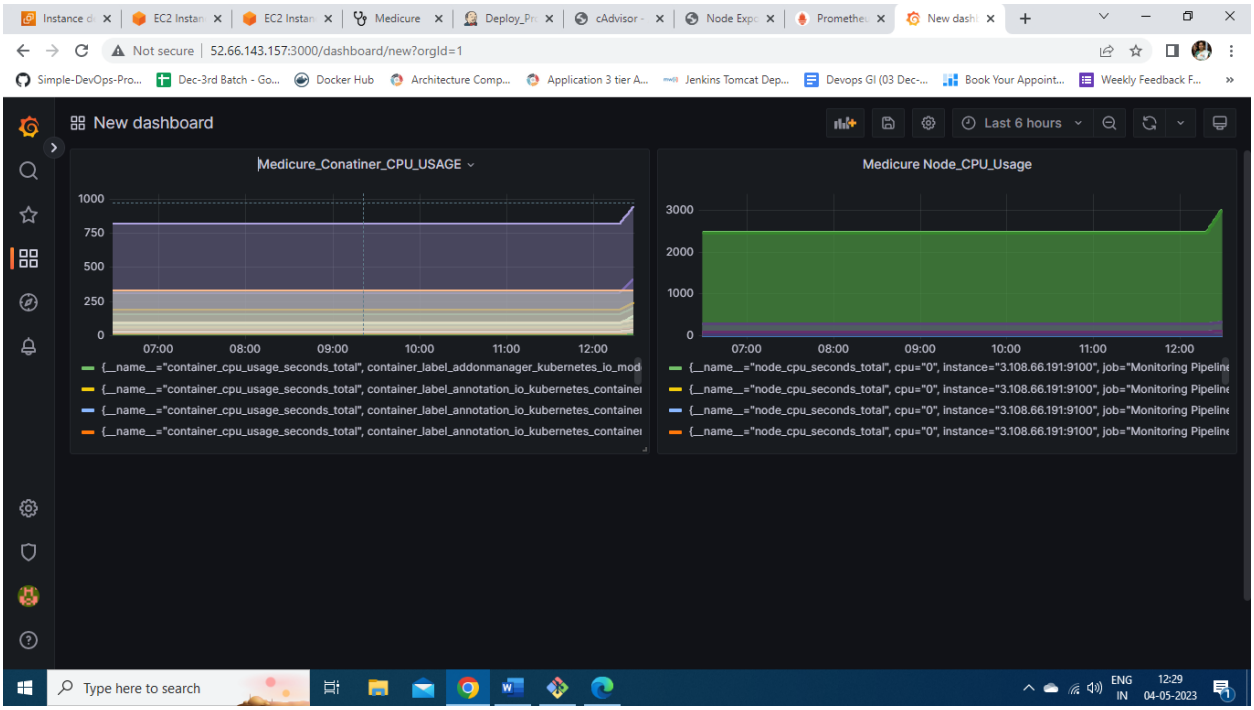


Here is graph showing the query container_cpu_uasge.





The dashboard for monitoring CPU usgae is shown here.



THANK YOU